NASA Conference Publication 2379

Computational Geometry and Computer-Aided Design

(NASA-CP-2379) COMPUTATIONAL GEOMETRY AND COMPUTER-AIDED DESIGN Extended Abstracts (NASA) 117 p HC A06/MF A01 CSCL 12A
N85-28563 THRU N85-28592 Unclas G3/59 20833

Abstracts of papers presented at a conference held at New Orleans, Louisiana June 5-8, 1985

NASA

NASA Conference Publication 2379

Computational Geometry and Computer-Aided Design

Compiled by
Temple H. Fay
University of Southern Mississippi
Hattiesburg, Mississippi

John N. Shoosmith
NASA Langley Research Center
Hampton, Virginia

Abstracts of papers presented at a conference sponsored by the National Aeronautics and Space Administration and the University of Southern Mississippi and held at New Orleans, Louisiana June 5–8, 1985

NASA
National Aeronautics and Space Administration

Scientific and Technical Information Branch

1985

PREFACE

This publication contains extended abstracts of papers presented at the International Conference on Computational Geometry and Computer-Aided Design, held in New Orleans, Louisiana, June 5-8, 1985.

With the continued growth of computer applications in design and manufacturing, increasingly powerful mathematical tools are being developed to solve scientific and engineering problems. Specifically, computer representation, analysis, and synthesis of shape information is a subject of intense development. Improvements are being made in the numerical representation of curves and surfaces, solid geometry modeling, management of geometric data, and development of geometric standards. This conference provides both invited and submitted papers in such areas, with the focus on geometric and mathematical aspects rather than on software aspects. The goal of the conference is to provide a forum where researchers and implementers from government, academia, and industry can discuss state-of-the-art developments and suggest problems and directions for future research.

This conference was conceived to be a followup to a symposium on computer-aided design held at NASA Langley Research Center in April 1983. Problems using three-dimensional geometry to model complex objects are of considerable importance at NASA. At the Langley Research Center, computer-aided design of aircraft and space vehicles and the solution to three-dimensional Navier-Stokes fluid flow equations in complex geometric regions are two such problems. Financial support for the conference was provided by the Analysis and Computation Division, NASA Langley Research Center, and the University of Southern Mississippi. The conference was organized into six sessions and included nine invited addresses.

This volume was prepared for publication through the efforts of the staff of the Research Information and Applications Division, NASA Langley Research Center. Use of trade names or manufacturers' names in this publication does not constitute endorsement, either expressed or implied, by the National Aeronautics and Space Administration.

Temple H. Fay
John N. Shoosmith
Conference Co-Organizers

PRECEDING PAGE BLANK NOT FILMED

CONTENTS

PREFACE iii

INVITED ADDRESSES

1. SHAPE PRESERVING SPLINE INTERPOLATION 1
   John A. Gregory

2. THE CONSTRUCTION OF CURVES AND SURFACES USING NUMERICAL OPTIMIZATION TECHNIQUES 9
   David R. Ferguson

3. MULTIVARIATE SPLINE ALGORITHMS FOR CAGD 15
   Wolfgang Boehm

4. A SURVEY OF PATCH METHODS 21
   R. E. Barnhill

5. WHEN IS A BERNSTEIN-BEZIER CURVE THE GRAPH OF A FUNCTION? 29
   Harry W. McLaughlin

6. FREE-FORM DESIGN IN SOLID MODELLING 31
   M. J. Pratt

7. BOUNDARY TO CONSTRUCTIVE SOLID GEOMETRY MAPPINGS: A FOCUS ON 2-D ISSUES 35
   Donald P. Peterson

8. THE DEFINITION AND COMPUTATION OF A METRIC ON PLANE CURVES: THE MEANING OF A "FACE" ON A GEOMETRIC MODEL 37
   James D. Emery

9. SOME NEGATIVE RESULTS IN N-SIDED PATCHES 45
   Malcolmb Sabin

SESSION I - CURVES AND SHAPE CONTROL

10. EXAMINATION OF THE CIRCLE SPLINE ROUTINE 47
    Ronald M. Dolin and Dwight L. Jaeger

11. CONVEX INTERPOLATING SPLINES OF ARBITRARY DEGREE 51
    Edward Neuman

12. A NATURAL BIAS APPROACH TO CARDINAL SPLINE CURVES 55
    G. Yates Fletcher and David F. McAllister

13. ADAPTING SHAPE PARAMETERS FOR CUBIC BEZIER CURVES 59
    David Isacoff and Michael J. Bailey

PRECEDING PAGE BLANK NOT FILMED

SESSION II - SURFACES

14. TRIANGULAR BERNSTEIN-BEZIER PATCHES, A SURVEY AND NEW RESULTS 65
Gerald Farin

15. A UNIFORM SUBDIVISION METHOD FOR TRIANGULAR BEZIER PATCHES 67
Kenneth I. Joy

16. AN INTUITIVE APPROACH TO GEOMETRIC CONTINUITY FOR PARAMETRIC CURVES AND SURFACES 71
Tony D. DeRose and Brian A. Barsky

17. URN MODELS AND BETA-SPLINES 77
R. N. Goldman

SESSION III - GRID GENERATION AND CONTOURING

18. PARAMETERIZATION IN GRID GENERATION 83
C. Wayne Mastin

19. CONTOURING TRIVARIATE DATA 87
Sarah E. Stead and George T. Makatura

20. GENERATION OF SURFACE COORDINATES BY ELLIPTIC PARTIAL DIFFERENTIAL EQUATIONS 91
Z. U. A. Warsi

21. A SURVEY OF COMPOSITE GRID GENERATION FOR GENERAL THREE-DIMENSIONAL REGIONS 95
Joe F. Thompson

SESSION IV - SOLID MODELLING

22. A UNIFIED REPRESENTATION SCHEME FOR SOLID GEOMETRIC OBJECTS USING B-SPLINES 99
Dennis Bahler

23. IMPROVING THE EFFECTIVENESS OF INTEGRAL PROPERTY CALCULATION IN A CSG SOLID MODELING SYSTEM BY EXPLOITING PREDICTABILITY 105
Alan L. Clark

SESSION V - SURFACES

24. SPLINE CURVES, WIRE FRAMES AND BVALUE 109
Leigh Smith and Frederick Munchmeyer

25. NUMERICAL ESTIMATION OF THE CURVATURE OF BIOLOGICAL SURFACES 113
P. H. Todd

26. A NEW TECHNIQUE FOR SYSTEM-TO-SYSTEM TRANSFER OF SURFACE DATA 117
Michael W. Sterling, Michael E. Lucius, and William J. Gordon

SESSION VI - CURVE INTERSECTION

27. CURVED FINITE ELEMENTS AND CURVE APPROXIMATION 119
M. Louisa Baart

28. COMPARISON OF TWO ALGEBRAIC METHODS FOR CURVE/CURVE INTERSECTION 123
Yves de Montaudouin and Wayne Tiller

29. A COMPARISON OF THREE CURVE INTERSECTION ALGORITHMS 127
Thomas W. Sederberg and Scott R. Parry

N85-28564

SHAPE PRESERVING SPLINE INTERPOLATION

John A. Gregory
Brunel University
Uxbridge, UK

1. INTRODUCTION

The spline-under-tension, developed by Schweikert[14] and Cline[2], introduces a parameter which gives some control on the shape of the spline curve. The tension spline involves the use of hyperbolic functions and can be considered within a general setting proposed by Pruess[13], where other alternatives are discussed. In particular a rational spline due to Späth[15] is considered.

For parametric representations, Nielson[11] describes a polynomial alternative to the spline-under-tension. Here use is made of the additional freedom given by relaxing C^2 parametric continuity to that of 'geometric' or 'visual' C^2 continuity. This idea has more recently been taken up by Barsky and Beatty[1].

The specific problem of shape preserving interpolation has been considered by a number of authors. Fritsch and Carlson[6] and Fritsch and Butland[5] discuss the interpolation of monotonic data using C^1 piecewise cubic polynomials. McAllister, Passow and Roulier[9] and Passow and Roulier[12] consider the problem of interpolating monotonic and convex data. They make use of piecewise polynomial Bernstein-Bezier representations and introduce additional knots into their schemes. In particular McAllister and Roulier[10] describe an algorithm for quadratic spline interpolation.

In this paper we will discuss a rational spline solution to the problem of shape preserving interpolation based on references 3, 4, 7 and 8. The rational spline is represented in terms of first derivative values at the knots and provides an alternative to the spline-under-tension. The idea of making the shape control parameters dependent on the first derivative unknowns is then explored. We are then able to preserve the monotonic or convex shape of the interpolation data automatically through the solution of the resulting non-linear consistency equations of the spline.

2. A RATIONAL ALTERNATIVE TO THE SPLINE-UNDER-TENSION

Let (x_i,f_i), $i = 1,\ldots,n$, be a given set of real data, where $x_1 < x_2 < \ldots < x_n$ and let d_i, $i = 1,\ldots,n$, denote first derivative values defined at the knots x_i, $i = 1,\ldots,n$. A function $s \in C^1[x_1,x_n]$ such that

$$s(x_i) = f_i \quad \text{and} \quad s^{(1)}(x_i) = d_i \ , \quad i = 1,\ldots,n \tag{2.1}$$

is piecewise defined for $x \in [x_i,x_{i+1}]$, $i = 1,\ldots,n-1$ by $s(x) = s_i(x;r_i)$, where

$$s_i(x;r_i) = \frac{(1-\theta)^2(1-\theta+r_i\theta)f_i+(1-\theta)^2\theta h_i d_i-\theta^2(1-\theta)h_i d_{i+1}+\theta^2(\theta+r_i(1-\theta))f_{i+1}}{1+(r_i-3)\theta(1-\theta)} \tag{2.2}$$

and

$$\theta = (x-x_i)/h_i \ , \quad h_i = x_{i+1}-x_i \ , \quad r_i > -1 \ . \tag{2.3}$$

The parameters r_i will be used to control the shape of the curve s. The case $r_i = 3$ is that of piecewise cubic Hermite interpolation.

The C^2 spline constraints

$$s^{(2)}(x_{i-}) = s^{(2)}(x_{i+}) \ , \quad i = 2,\ldots,n-1 \ , \tag{2.4}$$

give the consistency equations

$$\begin{aligned} h_i d_{i-1} + [h_i(r_{i-1}-1) + h_{i-1}(r_i-1)]d_i + h_{i-1}d_{i+1} \\ = h_i r_{i-1}\,\Delta_{i-1} + h_{i-1} r_i\,\Delta_i \ , \quad i = 2,\ldots,n-1 \ , \end{aligned} \tag{2.5}$$

where

$$\Delta_i = (f_{i+1}-f_i)/h_i \ , \quad i = 1,\ldots,n-1 \tag{2.6}$$

and we assume that d_1 and d_n are given as end conditions. The case $r_i = 3$ corresponds to that of cubic spline interpolation.

Assume that

$$r_i \geq r > 2 \ , \quad i = 1,\ldots,n-1 \ . \tag{2.7}$$

Then the linear system (2.5) is strictly diagonally dominant and hence has a unique solution. The solution is also bounded with respect to the r_i since

$$\max|d_i| \leq [r/(r-2)]\max|\Delta_i| \ . \tag{2.8}$$

The rational cubic (2.2) can be written as

$$s_i(x;r_i) = \ell_i(x) + e_i(x;r_i) \tag{2.9}$$

where ℓ_i is the linear interpolant

$$\ell_i(x) = (1-\theta)f_i + \theta\, f_{i+1} \tag{2.10}$$

and

$$e_i(x;r_i) = \frac{\theta(1-\theta)[(2\theta-1)(f_{i+1}-f_i) + (1-\theta)h_i d_i - \theta h_i d_{i+1}]}{1 + (r_i-3)\theta(1-\theta)} \tag{2.11}$$

Thus, as the parameters r_i are increased, it can be shown that the rational spline s converges uniformly to a piecewise defined linear interpolant. An identical argument applies to the rational spline representation of parametric curves.

Suppose $f_i = f(x_i)$ and $f_{i+1} = f(x_{i+1})$, where $f \in C^4[x_i x_{i+1}]$. Then an error bound for the rational cubic on $[x_i,x_{i+1}]$ is given by

$$|f(x) - s_i(x;r_i)| \leq \frac{h_i}{4c_i} \max\left\{|f_i^{(1)} - d_i| \ , \ |f_{i+1}^{(1)} - d_{i+1}|\right\}$$

$$+ \frac{1}{384c_i}\left[h_i^4 \| f^{(4)} \| + |r_i-3|\left\{\frac{1}{4} h_i^4 \| f^{(4)} \| + 4h_i^3 \| f^{(3)}\| + 12h_i^2 \| f^{(2)} \|\right\}\right] , \tag{2.12}$$

where

$$c_i = \min\{1,(1+r_i)/4\} \quad \text{and} \quad \|f\| = \max_{x \in [x_i,x_{i+1}]} |f(x)| .$$

This result will influence the choice of the parameters r_i when the interpolation data are monotonic or convex. In particular we wish to choose r_i such that $r_i - 3 = 0(h_i^2)$, whilst maintaining monotonicity or convexity of the interpolant.

3. THE INTERPOLATION OF MONOTONIC DATA

Suppose that

$$f_1 < f_2 < \ldots < f_n \quad \text{(monotonic increasing data)} \tag{3.1}$$

and assume the derivative values d_i satisfy the necessary monotonicity conditions

$$d_i > 0 , \quad i = 2,\ldots,n-1 , \tag{3.2}$$

where $d_1 > 0$ and $d_n > 0$ are given. Then

$$r_i = 1 + (d_i + d_{i+1})/\Delta_i \tag{3.3}$$

ensures that $s_i^{(1)}(x;r_i) > 0$ on $[x_i,x_{i+1}]$ and hence the rational cubic is monotonic increasing.

Substituting (3.3) into the C^2 consistency equations (2.5) gives the non-linear system

$$d_i[-c_i + a_{i-1}d_{i-1} + (a_{i-1} + a_i)d_i + a_i d_{i+1}] = b_i , \quad i = 2,\ldots,n-1 , \tag{3.4}$$

where

$$a_i = 1/(h_i\Delta_i) , \quad b_i = \Delta_{i-1}/h_{i-1} + \Delta_i/h_i , \quad c_i = 1/h_{i-1} + 1/h_i . \tag{3.5}$$

This system has a unique solution satisfying the monotonicity conditions (3.2). The "Gauss-Seidel" iteration

$$d_i^{(k+1)} = G_i(d_{i-1}^{(k+1)}, d_{i+1}^{(k)}), \quad i = 2,\ldots,n-1 , \tag{3.6}$$

where

$$G_i(\xi,\eta) = \frac{1}{2(a_{i-1}+a_i)}\{\zeta + [\zeta^2 + 4(a_{i-1}+a_i)b_i]^{\frac{1}{2}}\} , \tag{3.7}$$

$$\zeta = c_i - a_{i-1}\xi - a_i\eta , \tag{3.8}$$

provides a robust algorithm for solving (3.4), being globally convergent to the required positive solution. The method also provides an $O(h^4)$ accurate solution with reference to the error bound (2.12).

4. THE INTERPOLATION OF CONVEX DATA

Let

$$\Delta_1 < \Delta_2 < \ldots < \Delta_{n-1} \quad \text{(convex data)} , \tag{4.1}$$

and assume the derivative values satisfy the necessary convexity conditions

$$\Delta_{i-1} < d_i < \Delta_i , \quad i = 2,\ldots,n-1 , \tag{4.2}$$

where $d_1 < \Delta_1$ and $d_n > \Delta_{n-1}$ are given. Then

$$r_i = 1 + (d_{i+1} - \Delta_i)/(\Delta_i - d_i) + (\Delta_i - d_i)/(d_{i+1} - \Delta_i) \tag{4.3}$$

ensures that $s_i^{(2)}(x;r_i) > 0$ on $[x_i,x_{i+1}]$ and hence the rational cubic is convex.

The C^2 consistency equations are now

$$\left(\frac{\Delta_i - d_i}{d_i - \Delta_{i-1}}\right)^2 = \frac{h_i}{h_{i-1}}\left(\frac{d_{i+1} - \Delta_i}{\Delta_{i-1} - d_{i-1}}\right), \quad i = 2,\ldots,n-1 , \tag{4.4}$$

giving a non-linear system with a unique solution satisfying the convexity conditions (4.2). This solution can be found using a "Gauss-Seidel" iteration as in (3.6), where we now take

$$C_i(\xi,\eta) = \frac{h_{i-1}^{\frac{1}{2}}(\Delta_{i-1}-\xi)^{\frac{1}{2}}\Delta_i + h_i^{\frac{1}{2}}(\eta-\Delta_i)^{\frac{1}{2}}\Delta_{i-1}}{h_{i-1}^{\frac{1}{2}}(\Delta_{i-1}-\xi)^{\frac{1}{2}} + h_i^{\frac{1}{2}}(\eta-\Delta_i)^{\frac{1}{2}}} \tag{4.5}$$

and choose intial values $d_i^{(0)}$, $i = 2,\ldots,n-1$, for the iteration such that the residuals alternate. Finally, the convex spline method like the monotonic spline is $O(h^4)$ accurate.

REFERENCES

1. Barsky, B.A. and Beatty, J.C., Local control of bias and tension in Beta-Splines, ACM Trans. on Graphics, 2 (1983), pp.67-72.

2. Cline, A., Curve fitting in one and two dimensions using splines under tension, Comm. ACM, 17 (1974), pp.218-223.

3. Delbourgo, R. and Gregory, J.A., C^2 rational quadratic spline interpolation to monotonic data, IMA J. Numer. Anal., 3 (1983), pp.141-152.

4. Delbourgo, R. and Gregory, J.A., Shape preserving piecewise rational interpolation, to appear in SIAM J. Sci. Stat. Comput.

5. Fritsch, F.N. and Butland, J., A method for constructing local monotone piecewise cubic interpolants, SIAM J. Sci. Stat. Comput., 5 (1984), pp. 300-304.

6. Fritsch, F.N. and Carlson, R.E., Monotone piecewise cubic interpolation, SIAM J. Numer. Anal., 17 (1980), pp.238-246.

7. Gregory, J.A. and Delbourgo, R., Piecewise rational quadratic interpolation to monotonic data, IMA J. Numer. Anal., 2 (1982), pp.123-130.

8. Gregory, J.A., Shape preserving rational spline interpolation, in *Rational Approximation and Interpolation*, Graves-Morris, Saff and Varga, Eds., Springer-Verlag (1984), pp.431-441.

9. McAllister, D.F., Passow, E. and Roulier, J.A., Algorithms for computing shape preserving spline interpolations to data, Math. Comp., 31 (1977), pp.717-725.

10. McAllister, D.F. and Roulier, J.A., An algorithm for computing a shape preserving osculatory quadratic spline, ACM Trans. Math. Software, 7 (1981), pp.331-347.

11. Nielson, G., Some piecewise polynomial alternatives to splines under tension, in *Computer Aided Geometric Design*, Barnhill and Riesenfeld, Eds., Academic Press (1974), pp.209-236.

12. Passow, E. and Roulier, J.A., Monotone and convex spline interpolation, SIAM J. Numer. Anal., 14 (1977), pp.904-909.

13. Pruess, S., Alternatives to the exponential spline in tension, Math. Comp., 33 (1979), pp.1273-1281.

14. Schweikert, D., An interpolation curve using splines in tension, J. Math. and Phys., 45 (1966), pp.312-317.

15. Späth, H., *Spline Algorithms for Curves and Surfaces*, Utilitas Mathematica Pub. Inc., Winnipeg, (1974).

N85-28565

THE CONSTRUCTION OF CURVES AND SURFACES USING NUMERICAL OPTIMIZATION TECHNIQUES

DAVID R. FERGUSON
BOEING COMPUTER SERVICES
Seattle, Washington

INTRODUCTION:

Numerical optimization techniques are playing an increasing role in curve and surface construction. Often difficult problems in curve and surface construction, especially when some aspect of shape control is involved, can be phrased as a constrained optimization problem. Recently, efficient and robust optimization codes that allow both linear and non-linear constraints to be imposed on construction problems have become available (e.g., (1)). These codes enable a variety of such problems to be solved.

In this paper we explore four such classes of problems: parametric curve fitting with non-linear shape constraints; explicit surface fitting with linear shape constraints; surface fitting to scattered data giving rise to ill-posed problems; and, finally, variable knot problems. In each of these problems there is a non-linear aspect: either the shape of the curve or surface is important for manufacturing or engineering reasons or the shape affects the convergence of numerical algorithms which use the curve or surface or the placement of knots affects the accuracy of the fits.

In all cases the class of functions used is that of parametric spline curves and tensor or direct product spline surfaces. The reason for choosing this class is that splines provide flexible models that are easily evaluated and stored. Furthermore, the B-spline representation of splines leads to convenient expressions for shape control over regions.

We look first at the problem of constructing a curve satisfying shape constraints.

CURVE CONSTRUCTION WITH SHAPE CONSTRAINTS:

We begin our discussion with a quick review of shape for functions and curves including a critique of current methods. We then discuss the use of optimization methods to construct shape controlled curves.

PRECEDING PAGE BLANK NOT FILMED

In (2) the question of shape for parametric curves is studied in detail. There it is shown that the problem of constructing parametric curves $C(t) = (x(t),y(t))$ for which $C(t_i) = P_i$ for given planar points P_i and in addition requiring that the curve satisfy the inequality

(1) $x''(t)y'(t)-x'(t)y''(t) \geq 0$

for all t gives rise to optimization problems with non-linear constraints.

The first thing to be done is to reduce the continuous constraints defined by (1) to a finite set of discrete constraints. This is most conveniently done in the manner of (1) by settling on cubic splines as candidates for x(t) and y(t). By representing the spline curve as a B-spline series

(2) $C(t) = \sum Q_i B_i(t)$

the constraints (1) may be replaced by discrete constraints involving the B-spline coefficients Q_i. We describe this procedure and apply it to several examples.

The next two problems involve the construction of explicit tensor or direct product spline surfaces using optimization methods.

AN EXPLICIT SURFACE WITH LINEAR SHAPE CONSTRAINTS:

The previous example dealt with the problem of shape control for parametric curves and surfaces and gave rise to an optimization problem with non-linear constraints. We now turn to a class of problems involving the construction of explicit surfaces F(x,y) which approximate given surfaces but where the approximation is complicated by having a region where shape constraints need to be imposed.

The surfaces to be constructed are direct product spline surfaces of the type

(3) $\sum\sum a_{ij}B_i(x)C_j(y)$

and the type of approximation is least squares fitting. This is all straightforward. However, these approximations are to be used to describe properties of certain gases. These gas laws often have a region as in Figure 1 where the property is independent of one of the variables, e.g., the partial of temperature with respect to enthalpy is zero. Ignoring these 'flat' regions when doing the fitting gives rise to undesirable oscillations of the function. These oscillations severely affect the performance of the numerical integrators which will use this model.

However, it is possible using optimization methods to consider applying the constraints to the fitted function and thus to take advantage of the improved shape properties.

There are two problems to be considered here. The first is that of defining the region of constraints. Since the region has highly curved boundaries we approach this problem using the method of curved knot lines (3). The second problem consists of applying the constraints themselves. This we do by constraining the B-spline coefficients of the spline surface. We describe some experiments and show the results of using these methods.

ILL-POSED SURFACE PROBLEMS:

Another class of problems to which we will apply optimization techniques is that of surface fitting to scattered data. Again, consider the problem of fitting a direct product spline surface as described by (4) to data whose x-y coordinates are as depicted in Figure 2. There are good reasons for attempting a fit using direct product splines rather than using more traditional scattered data interpolation methods, e.g., those described in (5). The most important of these is simplification of the required data structures and evaluations.

On the other hand, there are problems. For instance, it will often happen that fitting with the direct product form (3) leads to ill-posed problems due to the fact that there are frequently elements $B_i(x)C_j(y)$ which are unsupported by the data. That is, the linear system which arises from (3) will be singular and these singularities must be dealt with. The proper handling of these singular systems has important consequences for the shape of the surface. For example one could solve the system by requiring that among all solutions the norm of the coefficient vector be minimized. This however, may lead to undesirable shape properties in the resulting surface. We will explore methods of handling these shape problems.

VARIABLE KNOT PROBLEMS:

The final class of problems that are candidates for optimization is the variable knot problems, especially those in two and three dimensions. For these problems, the proper placement of the knots is often of critical importance to the approximation. We will look at using optimization techniques to obtain best positions for the knots.

REFERENCES:

1. Gill, P.E., W. Murray, M. Saunders and N.H. Wright, User's Guide for NPSOL, Report SOL 84-7, Sept. 1984, Department of Operations Research, Stanford University, Stanford, California.

2. Ferguson, D.R. and A.K. Jones, Parametric Curve Fitting with Shape Control (submitted to SIAM J. Sci. Stat. Comp.)

3. Hayes, J.G., Curved Knot Lines and Surfaces with Ruled Segments, Dundee Conf. on Numerical Analysis, 1981, pg 140 - 156.

4. Gill, P.E., W. Murray and M.H. Wright, Practical Optimization, Academic Press, 1981, London and New York.

5. Franke, R and G.M. Nielson, A Method for Construction of Surfaces Under Tension, Rocky Mountain J. Math 14, No. 1, 1984, pp 203, 221.

ORIGINAL PAGE IS
OF POOR QUALITY

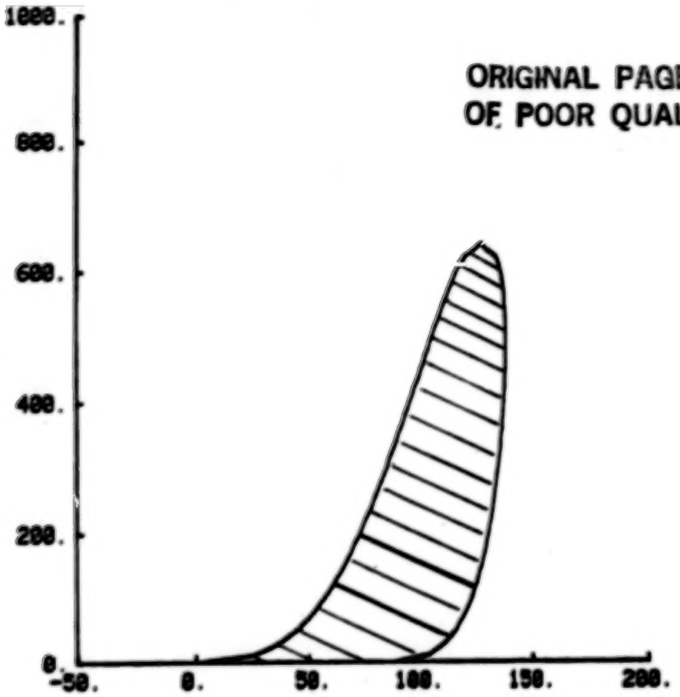


Figure 1
Constrained region for FREON 22.

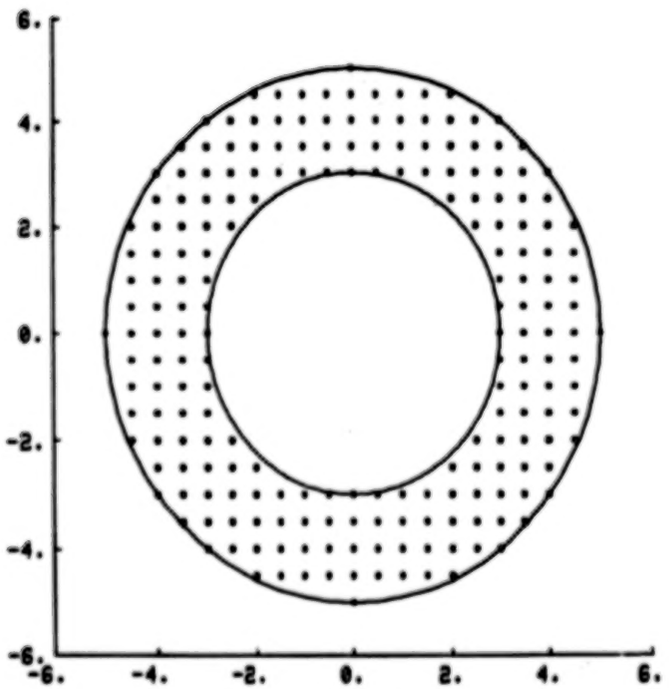


Figure 2
Domain leading to an ill-posed problem.

N85-28566

MULTIVARIATE SPLINE ALGORITHMS FOR CAGD

Wolfgang Boehm
Angewandte Geometrie und GDV
T. U. Braunschweig

ABSTRACT

The theory of multivariate splines was developed in the early 1980s, and since then has generated increasing interest. Researchers in computer-aided geometric design (CAGD) hoped to get a new, useful tool for the representation and handling of surfaces. This interest in multivariate splines is based on three fruitful ideas:

Schoenberg's geometric definition of splines (Schoenberg, 1966)

Geometric recursion and subdivision

The Bernstein-Bézier representation

Multivariate splines are easily developed from the geometric definition of splines, given by Schoenberg in 1966: A polyhedron P that spans R^m is affinely mapped into R^s, $s < m$. All points $p \in P$ that are mapped onto a single point $x \in R^s$ from an m-s dimensional polyhedron Q(x). The volume V(x) of Q is called a spline function. It is a piecewise polynomial of order m-s and "smooth". Note that one can easily find different polyhedra defining the same spline.

The image of a vertex of P will be called a knot, while the image of an edge of P will be called a knot line, connecting two knots. These knot lines correspond to the segment boundaries.

Two special polyhedra present themselves for the definition of B-splines: a simplex S and a box or parallelepiped B, where the edges of S project into an irregular grid, while the edges of B project into the edges of a regular grid, as shown in figure 1. More general splines may be found by forming linear combinations of these B-splines, where the three-dimensional coefficients are called the spline "control points".

Note that univariate splines are simplex splines, where s = 1, whereas splines over a regular triangular grid are box splines, where s = 2.

Two simple facts render the development of the construction of B-splines:

Any "face" of a simplex or a box is again a simplex or box but of lower dimension.

PRECEDING PAGE BLANK NOT FILMED

Any simplex or box can be easily subdivided into smaller simplices or boxes.

The first fact gives a geometric approach to Mansfield-like recursion formulas that express a B-spline in B-splines of lower order, where the coefficients depend on x. By repeated recursion, the B-spline will be expressed as B-splines of order 1; i.e. piecewise constants. Considering the corresponding "nets" of control points, one gets de Boor - like algorithms for the calculation of a given linear combination of B-splines at x.

In the case of a simplex spline, the second fact gives a so-called "insertion algorithm" that constructs the new control points if an additional knot is inserted. In the case of a box spline, this fact gives the so called "subdivision" algorithm, which constructs a "refinement" of the control net (see figure 2).

If more than one vertex of a simplex is mapped into one knot, the corresponding spline function will be degenerate. In particular, if the knots form a simplex of R^s, the spline will be a "truncated Bernstein polynomial". In this case the Mansfield-like recursion formula for simplex splines degenerates to the well-known recursion formula for Bernstein polynomials, while the de Boor - like algorithm degenerates to the algorithm of de Casteljau, where the control points are called Bézier points. Clearly, the Bernstein expansion of a B-spline corresponds to a suitable simplicial decomposition of the simplex or box, such that the Bernstein representation of a B-spline can easily be constructed (see figure 3). Furthermore, especially if the B-spline is a truncated Bernstein polynomial, one gets the subdivision algorithm for Bernstein polynomials as well as a refinement of the Bézier net.

Fortunately, two of the above algorithms seem to fall in the class of non-tensor-product box splines:

The approximation of the spline by a repeated refinement for global representation

The construction of the Bézier net for application of the full Bernstein-Bézier method for local representation

It should be mentioned that both algorithms start from the control net and use - in the known cases - repeated "filling and/or averaging" procedures, as shown in figure 4 for the case of a triangular grid:

A rhombic scheme is filled with data from the previous step,

A new scheme is formed by line averaging these data.

REFERENCES

Boehm, W. (1983): Subdividing Multivariate Splines, CAD 15, 345-352.

Boehm, W. (1984): Calculating with Box Splines, CAGD 1, 149-162.

Dahmen, W., and Micchellli, C. (1983): Multivariate Splines - A New Constructive Approach. R. Barnhill and W. Boehm, Surfaces in CAGD, edited by North-Holland, Amsterdam.

Dahmen, W., and Micchelli, C. (1984): Subdivision Algorithms for the Generation of Box Spline Surfaces, CAGD 1, 115-129.

Prautzsch, H. (1984): Unterteilungsalgorithmen fuer multivariate Splines - ein geometrischer Zugang. Ph.D. Thesis, Technische Universitaet Braunschweig.

Prautzsch, H. (1985): Generalized Subdivisioin and Convergence, CAGD, to appear.

Schoenberg, I. J. (1966): On Spline Functions. University of Wisconsin MRC-TSR-625, Madison.

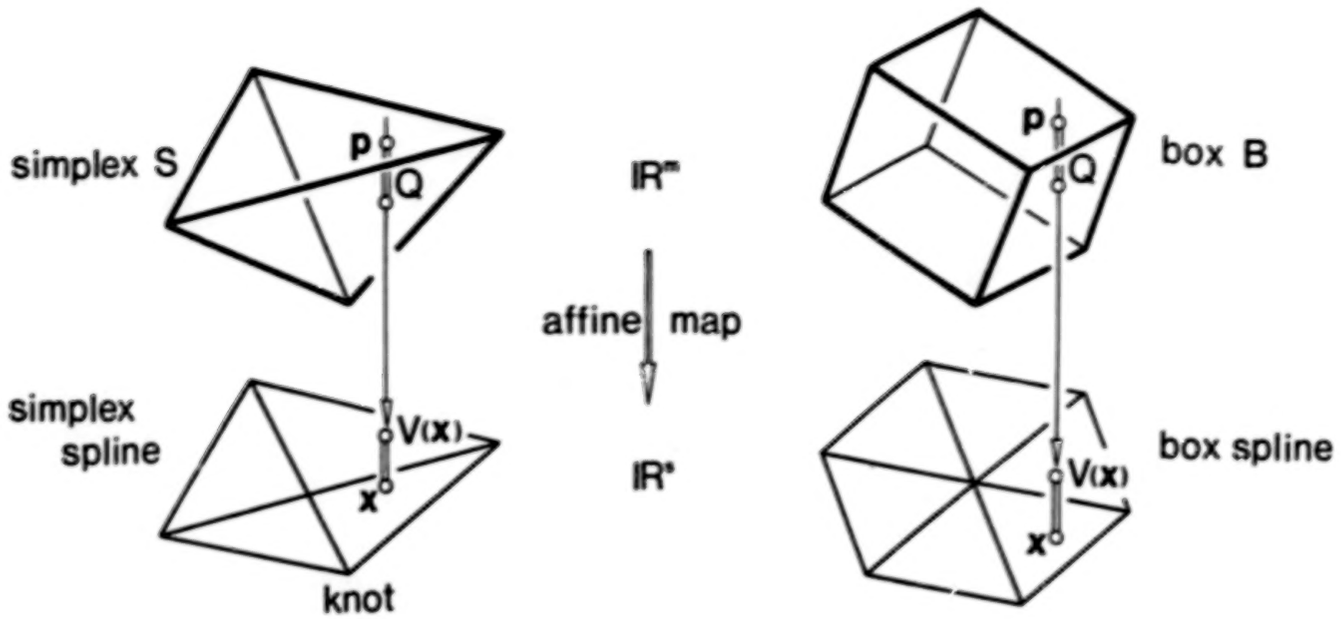


FIGURE 1

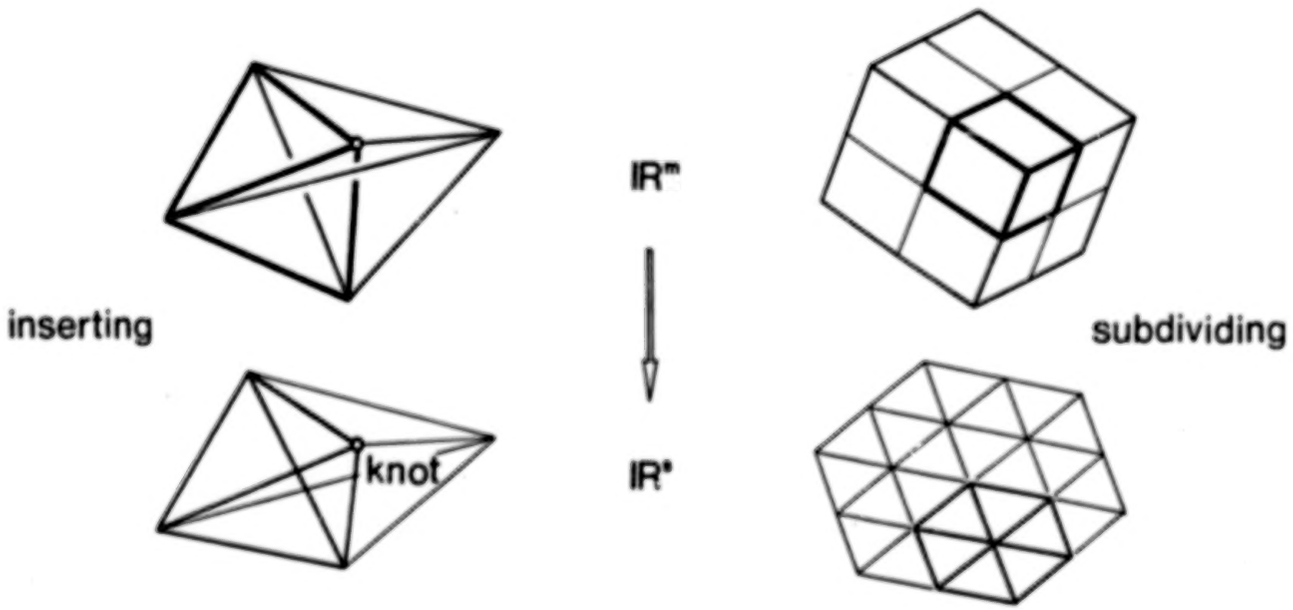


FIGURE 2

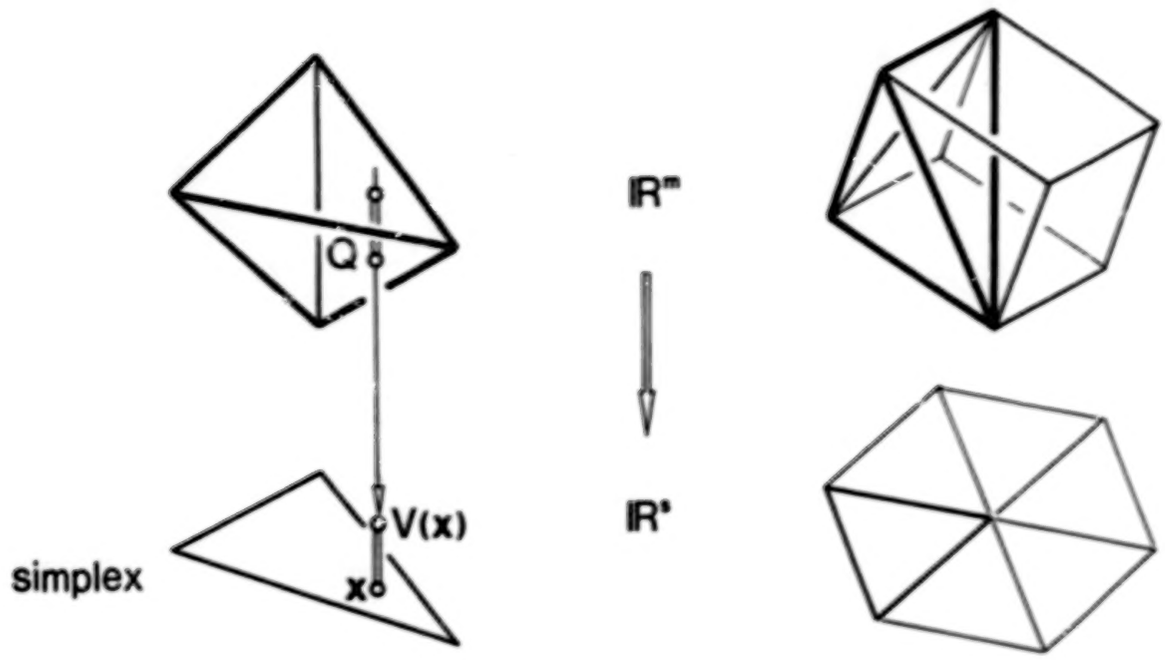


FIGURE 3

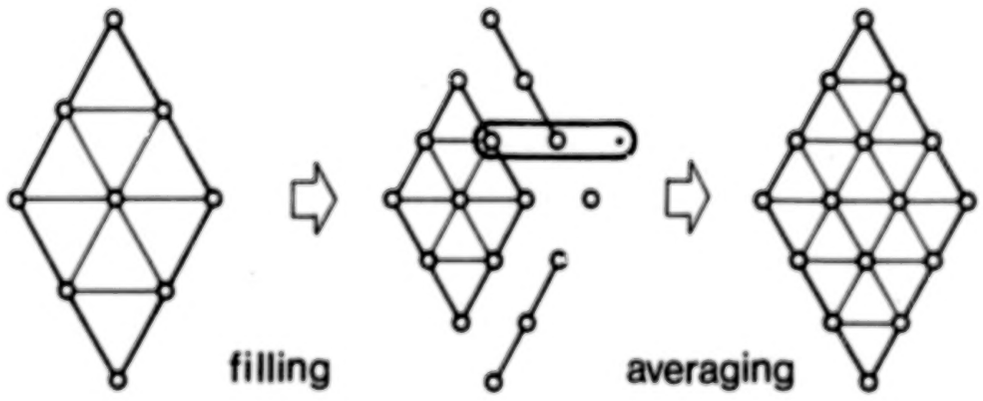


FIGURE 4

N85-28567

A Survey of Patch Methods

R. E. Barnhill
Department of Mathematics
University of Utah
Salt Lake City, Utah 84112

Introduction

There are two broad classes of interpolation methods for surfaces: patch methods (Barnhill, 1985) and distance-weighted methods. We discuss patch methods in this paper. (Readers interested in distance-weighted methods should consult Barnhill and Stead (1984) and Franke (1982) and the references therein.) Patch methods are somehow a response to the fact that surface geometry is local, that is, only small parts of a surface are created at a time. We present the two categories of patches, transfinite patches and finite dimensional patches, followed by a discussion of trivariate patches. We do not discuss how to create the domains for patches nor data that are needed for the various schemes. Information on creating triangles and tetrahedra is given in Barnhill and Little (1984) and the references therein and information on creating gradient data in Stead (1984) and the references therein.

PRECEDING PAGE BLANK NOT FILMED

Transfinite Patches

In the car industry one has whole curves of information to interpolate. S. A. Coons (1964) fixed his attention on one rectangular set of four curves and constructed interpolants to position and cross-boundary derivative information all along the curves. W. J. Gordon (1969) called such interpolation "transfinite" because entire curves of data are interpolated. R. E. Barnhill, J. A. Gregory, and L. E. Mansfield noticed that the C^1 Coons patch did not interpolate to the cross-boundary derivative on two of the four curves. (Gregory, 1974; Barnhill, 1977).

The difficulty was traced to the lack of commutativity in the (1,1) derivatives (the "twists") in the Coons patch. Shortly thereafter Barnhill and Gregory (1975) created "compatibility correction" terms which introduced a variable twist, that is, both of the two implied twists are included.

The idea of transfinite triangular patches was initiated by Barnhill, Birkhoff, and Gordon (1973). They created a C^1 triangular Coons patch on the standard triangle with vertices (1,0) (0,1) and (0,0). For several years attempts were made to generalize these patches to an arbitrary triangle; one successful effort is in Klucewicz' (1977) thesis.

Barycentric coordinates are natural for triangles, and transforming between cartesion xy-coordinates and the barycentric coordinates b_i is defined by the equation

$$(1) \qquad (x,y) = \sum_{i=1}^{3} b_i V_i \qquad\qquad 1 = \sum_{i=1}^{3} b_i$$

where the V_i are the vertices of the triangle.

This equation has a solution if and only if the triangle has positive area, i.e., the V_i are not collinear. Barycentric coordinates are very useful in working with triangles which have been used by the finite element engineers for a long time. (Finite elements enter the story below.) Finally, F. F. Little proposed a barycentric calculus which enables us to take a derivative with respect to a given direction. More precisely, if $F = F(x,y)$, then F can be rewritten in terms of barycentric coordinates by substituting equations (1) for x and y: call $F(x,y) = F(b_1V_1 + b_2V_2 + b_3V_3) = G(b_1,b_2,b_3)$. Then we can take derivatives of F with respect to a direction e by use of the chain rule:

$$\frac{\partial F}{\partial e} = \sum_{i=1}^{3} \frac{\partial G}{\partial b_i} \frac{\partial b_i}{\partial e}$$

The Barnhill, Birkhoff, and Gordon interpolants for an arbitrary triangle have been used at Utah for some time, but have only recently been published: A C^1 BBG scheme (and its "discretization", defined below) is in Barnhill (1983). A C^2 BBG scheme is in Alfeld and Barnhill (1984). A bivariate BBG scheme, a trivariate BBG scheme, and a bivariate "radial Nielson" interpolant together with their discretizations are in Barnhill and Little (1984). (Trivariate patches are discussed below.) Other transfinite triangular interpolants are Gregory's (1983) symmetric schemes.

Finite Dimensional Patches

For some time engineers in finite element analysis have used piecewise polynomials defined over rectangles or triangles. (Strang and Fix, 1973; Zienkiewicz, 1977) These polynomials are the basis functions for interpolation schemes. However, the finite element method is used to calculate what we would call the positions and so the cardinal form of the interpolant, that is, the form in which the data functionals occur explicitly, is not needed. Thus, for example, the well-known 18 degree of freedom C^1 quintic's cardinal form was discovered only recently by Barnhill and Farin (1981) although the scheme has been used in finite element calculations for years.

Kolar, Kratochvil, Zenisek, and Zlamal (1971) discuss a "hierarchy" of polynomial interpolants defined over triangles. They give precise statements on the degree of the polynomial needed to obtain a certain global smoothness when the polynomial scheme is applied over each triangle in a network, for example, linear C^0, quintic C^1, nonic C^2, etc. Recently at Utah Whelan (1985) and others have developed cardinal forms for the C^2 nonic.

The finite element engineers have developed a second type of piecewise polynomial where a given triangle is subdivided and a piecewise scheme is defined over the subdivisions. In order to distinguish these two types of triangular schemes, we call the first type "macro" triangles and the second type (subdivision) "micro" triangles. The best known microtriangle scheme is the cubic C^1 Clough-Tocher element for which the macrotriangle is subdivided at its centroid into three microtriangles. Barnhill, Farin, and Little worked on creating a C^2 quintic Clough-Tocher interpolant with the macrotriangle subdivided into seven microtriangles, but have not yet succeeeded. Alfeld

(1985) has created such a scheme over nine microtriangles.

An important development for the discovery and description of piecewise polynomial schemes over triangles is Farin's (1980) generalization of the "Bernstein-Bezier" methods to arbitrary triangles. There are two levels of generalization here, since Bezier's original development was for rectangles. Sabin (1976) described triangular patches over equilateral triangles. Bernstein-Bezier methods are applicable to any piecewise polynomial scheme because they are one representation of the polynomial. The fact that geometric information can be fairly easily determined from this representation gives the methods its power.

Transfinite patches can be specialized to finite dimensional patches by discretizing the transfinite data, for example, by replacing a curve by its (univariate) cubic Hermite interpolant. The "serendipity elements" of the engineers can be viewed as examples of the discretization of transfinite patches.

A very recent idea is that of "visual continuity". C^1 visual continuity (VC^1) means tangent plane continuity. The concept becomes important when fitting together patches whose domains do not "match" such as a triangle and a rectangle. By not matching we mean that, for example, the isoparametric lines from a rectangle do not correspond to any standard lines in a triangle. Gregory and Charrot (1980) fit a triangular patch into a network of rectangular patches in a VC^1 way. Visual continuity has been discussed for triangular patches by Herron (1985) and for triangular and rectangular Bezier patches by Farin (1982).

Trivariate Patches

There are many applications which involve the creation of a surface in four-space, that is, a trivariate function. We mentioned above that trivariate BBG interpolants over tetrahedra are given by Barnhill and Little (1984). Another transfinite interpolant is formed by means of a convex combination of BBG projectors by Alfeld (1984a). A C^1 Clough-Tocher-like tetrahedral interpolant, which is quintic over four subtetrahedra and requires C^2 data, is presented by Alfeld (1984b). A second C^1 tetrahedral interpolant, which is cubic over twelve subtetrahedra and requires C^1 data, has just been announced by Worsey and Farin (manuscript in preparation). In fact this is a special case of their n-dimensional simplicial interpolants. Gregory (1985) has generalized his symmetric triangular interpolant (a convex combination of projectors) to n-dimensional simplices.

Acknowledgements

This presentation has been improved by friendly interactions with the Math CAGD Group at Utah and, in particular, with A. J. Worsey.

This research was supported in part by the Department of Energy under contract DE-AC02-82ER12046 to the University of Utah.

References

R. E. Barnhill (1985), Surfaces in computer aided geometric design: a survey with new results, Computer Aided Geometric Design (to appear).

R. E. Barnhill and S. E. Stead (1984), Multistage trivariate surfaces, Rocky Mountain Journal of Mathematics, 14, 103-118.

R. H. Franke (1982), Scattered data interpolation: tests of some methods, Mathematics of Computation, 38, 181-200.

R. E. Barnhill and F. F. Little (1984), Three- and four-dimensional surfaces, Rocky Mountain Journal of Mathematics, 14, 77-102.

S. E. Stead (1984), Estimation of gradients from scattered data, Rocky Mountain Journal of Mathematics, 14, 265-280.

S. A. Coons (1964), Surfaces for computer aided design, Mechanical Engineering Department, M.I.T., revised, 1967. (Available as AD 663 504 from the National Technical Information Service, Springfield, VA 22161.)

W. J. Gordon (1969), Distributive lattices and the approximation of multivariate functions, in: I. J. Schoenberg, editor, Approximations with Special Emphasis on Splines, University of Wisconsin Press, Madison.

J. A. Gregory (1974), Smooth interpolation without twist constraints, in: R. E. Barnhill and R. F. Riesenfeld, editors, Computer Aided Geometric Design, Academic Press, New York.

R. E. Barnhill (1977), Representation and approximation of surfaces, in: J. R. Rice, editor, Mathematical Software III, Academic Press, New York.

R. E. Barnhill and J. A. Gregory (1975), Polynomial interpolation to boundary data on triangles, Mathematics of Computation, 29, 726-735.

R. E. Barnhill, G. Birkhoff, and W. J. Gordon (1973), Smooth interpolation in triangles, Journal of Approximation Theory, 8 (2), 114-128.

I. M. Klucewicz (1977), A piecewise C^1 interpolant to arbitrarily spaced data, Masters thesis, Department of Mathematics, University of Utah, Salt Lake City.

R. E. Barnhill (1983), Computer aided surface representation and design, in: R. E. Barnhill and W. Boehm, editors, Surfaces in Computer Aided Geometric Design, North-Holland, Amsterdam.

P. Alfeld and R. E. Barnhill (1984), A transfinite C^2 interpolant over triangles, Rocky Mountain Journal of Mathematics, 14, 17-40.

J. A. Gregory (1983), C^1 rectangular and non-rectangular surface patches, in: R. E. Barnhill and W. Boehm, editors, Surfaces in Computer Aided Geometric Design, North-Holland, Amsterdam.

G. Strang and G. J. Fix (1973), An Analysis of the Finite Element Method, Prentice-Hall, Englewood Cliffs, N. J.

O. C. Zienkiewicz (1977), The Finite Element Method, McGraw-Hill, London.

V. Kolar, J. Kratochvil, A. Zenisek, and M. Zlamal (1971), Technical, Physical, and Mathematical Principles of the Finite Element Method, Academia, Czechoslovak Academy of Sciences, Prague, Czechoslovakia.

R. E. Barnhill and G. Farin (1981), C^1 quintic interpolation over triangles: two explicit representation, International Journal for Numerical Methods in Engineering, 17, 1763-1778.

T. Whelan (1985), A representation of a C^2 interpolant over triangles, submitted for publication, Computer-Aided Geometric Design.

P. Alfeld (1985), A bivariate C^2 Clough-Tocher scheme, Computer Aided Geometric Design, (to appear).

G. Farin (1980), Bezier polynomials over triangles and the construction of piecewise C^r polynomials, TR/91, Department of Mathematics, Brunel University, Uxbridge, England.

M. A. Sabin (1976), The use of piecewise forms for the numerical representation of shape, Tanulmanyok 60/1977, Hungarian Academy of Sciences, Budapest, Hungary.

J. A. Gregory and P. Charrot (1980), A C^1 triangular interpolation patch for computer aided geometric design, Computer Graphics and Image Processing, 13, 80-87.

G. J. Herron (1985), Smooth closed surfaces with discrete triangular interpolants, Computer Aided Geometric Design (to appear).

G. Farin (1982), A construction for the visual C^1 continuity of polynomial surface patches, Computer Graphics and Image Processing, 20, 272-282.

P. Alfeld (1984a), A discrete C^1 interpolant for tetrahedral data, Rocky Mountain Journal of Mathematics, 14, 5-16.

P. Alfeld (1984b), A trivariate Clough-Tocher scheme for tetrahedral data, Computer Aided Geometric Design, 1, 169-181.

J. A. Gregory (1985), Interpolation to boundary data on the simplex, Computer Aided Geometric Design (to appear).

N85-28568

WHEN IS A BERNSTEIN-BÉZIER CURVE THE GRAPH OF A FUNCTION?

Harry W. McLaughlin
Rensselaer Polytechnic Institute
Troy, New York

In this paper we consider the question of determining when a Bernstein-Bézier cubic curve in the plane can be represented as the graph of function in some fixed orthogonal coordinate system.

One lets θ be an arbitrary point in the plane and D and N be nonzero orthogonal vectors in the plane. Then every point P in the plane can be written in the form $\theta + uD + vN$ for a unique choice of scalars u and v. The triple (θ,D,N) is an orthogonal coordinate system and the coordinates of P in this coordinate system are u and v. If for $t_1 < t < t_2$, $t_1 < t_2$,

$$B_3(t) = \sum_{k=0}^{3} P_k \binom{3}{k} t^k(1-t)^{3-k}$$

where the P_k's are given points in the plane, then one can also write

$$B_3(t) = \theta + u(t)D + v(t)N, \quad t_1 < t < t_2$$

for some functions u(t) and v(t). The curve $B_3(t)$ is said to be the graph of a function in the coordinate system (θ,D,N) if v(t) can be written

$$v(t) = f(u(t)), \quad t_1 < t < t_2$$

for some real valued function f defined on the range of u. The function v(t) has such a representation if and only if: whenever u(t) takes on the same value for two distinct values of t, then so does v(t).

To aid in the analysis one can introduce the notion of a curve being monotone in a given direction. The curve, $B_3(t)$, $t_1 < t < t_2$, is monotone in the direction D if whenever $t_1 < \alpha < \beta < t_2$, one has $(B_3(\beta)-B_3(\alpha))\cdot D > 0$. If $B_3(t)$ is monotone in the direction, D, then $B_3(t)$ is the graph of a function in the coordinate system (θ,D,N). However, the converse is not true in general.

We classify those curves of the form, $B_3(t)$, which can be represented as the graph of a function in the coordinate system (θ,D,N), by studying separately those which are monotone in the direction $\pm D$ and those which are not.

Those curves, $B_3(t)$, which are not monotone in the directions $\pm D$ and which are graphs of functions in (θ,D,N), are necessarily straight lines.

For the other case one has, as a preliminary theorem: the curve, $B_3(t)$, $t_1 \leq t \leq t_2$, is monotone in the direction D if and only if for each $t_1 < t_0 < t_2$, either

(1) $B_3'(t_0) \cdot D > 0$

or

(2) $B_3'(t_0) = 0$ and $B_3''(t_0) \neq 0$ and $B_3''(t_0) \cdot D = 0$

or

(3) $B_3'(t_0) \cdot D = 0$ and $B_3'(t_0) \neq 0$

and

$$B_3''(t_0) = \lambda B_3'(t_0) \text{ for some scalar } \lambda$$

and

$$B_3'''(t_0) \neq \mu B_3'(t_0) \text{ for all scalars } \mu$$

The goal of the study has been to translate these analytic statements into geometric constraints on the control points, P_k's, of the curve $B_3(t)$. Necessary and sufficient conditions are generated by first studying necessary and sufficient conditions on the control points for cusps and then for inflection points, since each of these phenomena has to be dealt with to obtain the final theorem. The analysis is quite lengthy and will be reported on (in print) elsewhere.

This research has disclosed several unsolved elementary interpolation problems, each of which appears to be important.

Finally, it is noted that the notion of monotone in a direction D may be useful to those who study shapes of curves and surfaces since it is a way to describe the empirical notion that a given curve appears to be "headed in a given direction."

N85-28569

FREE-FORM DESIGN IN SOLID MODELLING

M.J. Pratt
Cranfield Institute of Technology
Cranfield, Bedford, England

Surface modelling techniques have a long history, which can be traced back well before the beginning of the computer era. Their origins are rooted in the aircraft, automotive and shipbuilding industries. Solid modelling is a more recent art, dating back only fifteen years or so, and was originally developed as a means of representing the shapes of components used in the less specialised mechanical engineering industries.

At first the two types of systems were developed largely independently of each other, and used very different techniques. Surfaces were represented in terms of parametric geometry capable of representing very general free-form shapes. Solids were modelled in terms of set operations involving simple volumetric primitives such as blocks, cylinders and cones, whose surfaces were usually represented by more classical implicit equations. In recent years, strenuous efforts have been made to bring together the virtues of both approaches, with varying degrees of success. Several commercially available solid modellers and even more research-oriented systems have now reached the stage where they can model solids with free-form surfaces; in some cases parametric geometry is used exclusively, while in others there is mixed use of parametric and implicit geometry.

From the designer's point of view the methods available to him for designing solids with free-form geometry are not ideal. They fall into three main classes, as follows:

1) Methods based largely on 'traditional' surface modelling techniques such as, for example, the construction of objects whose surfaces interpolate families of specified cross-sectional curves. Such methods are fine for aircraft fuselages and ship hulls but not easily applicable for the design of cylinder blocks or gearbox casings (see, for example, Tiller (1983)).

2) Methods using techniques which have proved relatively easy to implement in solid modellers of the boundary representation type. These are based, broadly speaking, on the initial definition of an object composed of blocks, cylinders and so on, followed by the use of a separately defined free-form surface to modify the object in some way. The object may be cut into two sections by the surface, or one or more of the object faces may be moved to lie on the new surface, for example. Objects defined in this manner usually have sharp edges uncharacteristic of many typical engineering components, whose geometry exhibits subtle blends and fillets stemming traditionally from the art of the mould-maker or pattern-maker. Methods of the type described are implemented, for example, in BUILD (Anderson, 1983).

3) Methods based on techniques which have proved relatively easy to implement in solid modellers of the constructive solid geometry (CSG) type. Here a consensus seems to be emerging that 'swept volume' solids provide a useful design method, based on the use of the volumes swept out by the motion of a simple primitive such as a sphere along a specified path. The volume of the sphere may vary, and the path may be curvilinear. This permits a wide range of interesting shapes to be generated, and has a useful application for defining surfaces of the 'rolling ball' type for filleting internal corners. As a general design method these techniques have limited application, however. References to this type of approach include Rossignac and Requicha (1984), van Wijk (1984). A fuller survey of all these methods is given in Várady and Pratt (1985).

The theme of this paper is that what is really needed is a much closer synthesis of parametric surface techniques with conventional solid modelling techniques. There are many engineering components whose gross geometry can be approximated in terms of the primitive volumes customarily provided such as blocks, cylinders, cones, etc., but whose precise modelling requires changes of a local nature involving the 'sculpturing' of edges, the rounding of internal corners and the construction of general blending surfaces over limited regions. It is not appropriate to try to construct the required free-form surfaces as separate entities and then to graft them onto the solid model, since the boundary constraints which must be applied in practical cases are actually determined by the model. The most satisfactory approach appears to be to use the gross model as a framework, and to provide means for modifying the surfaces composing its boundary in a free-form manner. This appears to imply the use of a modeller of the boundary representation type, but in fact there is no reason in principle why the bounding surfaces of primitive volumes in a CSG system may not be similarly dealt with. However, since the writer's own leanings are towards boundary representation the ideas which follow will be outlined in that context.

Consider first a cube, whose faces we will consider to be labelled 'front', 'back', 'left', 'right', 'top', and 'bottom'. Most boundary representation modellers will represent the plane surfaces containing the faces in some implicit manner, though parametric representations are used in a few cases. For our purposes the most convenient formulation is in terms of Bézier or B-spline surfaces. For example, if a uniform 3x3 grid is imposed upon the top face of the cube then the 4x4 set of resulting mesh nodes could be considered to be the control points of a bicubic Bezier representation of this plane face. So far, there is no geometric change in the cube, but at this stage it becomes possible to modify the geometry of the top face by moving the control points. The possibilities are

(i) manipulation of the four interior control points. This permits the creation of a bulge or a depression, for example, and leaves the four edges of the face unaltered.

(ii) manipulation also of the boundary control points. Here a limitation must be imposed; the movement must be constrained to lie in the planes of the adjoining faces. This ensures that the modified edge curves still lie in the planes of these faces, whose associated geometry has so far not been redefined.

Now suppose that the front face is also redefined as a Bézier patch. One set of four control points will be common to both the top and the front face, and these lie along the shared edge. It is therefore possible to blend this edge and obtain C^1 continuity between the two faces simply by relocating these four control points so that each lies midway between its immediate neighbors on the top and front faces respectively. Similarly, we could round off the corners of the cube completely by redefining the side faces as Bézier patches and adjusting the positions of the shared corner control points appropriately.

The types of procedure described have a number of advantages. In particular, the initial undeformed Bézier representation is set up automatically using the original model as a framework, and also in the examples given there are no toplogical changes in the model resulting from the deformation. The computational overheads are few, and it is worthy of note that no surface/surface intersection curves need to be computed. On the other hand, a practical implementation will require careful attention to the facilities available for the user to manipulate Bézier control points. He will probably need to work with whole lines or blocks of points for some purposes, though fine tuning of his surfaces will also require manipulation of single points.

Finally, it must be noted that certain refinements of the basic procedure are necessary to make it applicable in more general cases. For example, if it is desired to modify a cylindrical face then an exact parametric representation of the initial face will require the use of a rational form (Faux & Pratt, 1979). The same is true for the other simple quadric surfaces and the torus, all of which may be represented as rational biquadratics. In practice it will probably be better to use an equivalent formulation of higher degree, however, to give more freedom in imposition of boundary constraints on the deformed surfaces.

Next consider the situation where the modification to a particular face must be local to one particular edge, the remainder of the face remaining undeformed. This is easily achieved by the initial splitting of the original face into two by the insertion of a new edge, which will in many cases be parallel to the edge whose local region is to be modified. It will also be necessary to interpolate a new edge in this way if the face to be modified has more or less than four edges, since it is not then possible to parametrise the entire face in a natural way as a four-sided Bézier patch.

It is also simple to modify any four-sided region which can be expressed as a Bézier patch and which lies totally interior to a face of the model. The region to be modified must first be defined as a new face, and a parametrisation imposed upon it as described. Manipulation of the geometry of the new face is by means of control points as previously, but the use of a formulation of higher than cubic degree will probably be advantageous in this case. Then there will be enough freedom to allow the modified face to retain tangency across its edges with the surface of the original face if desired. It must be pointed out that the achievement of such results using Boolean operations is fraught with difficulties arising from the resolution of tolerance problems associated with tangencies of surfaces between the separate objects involved.

To summarise, the method suggested for the free-form modification of solid models has several important advantages. In particular, it avoids the use of 'detached' surfaces, Boolean operations and surface intersection computations.

It involves, at worst, only minor topological changes to the model, and will therefore be computationally efficient. The ideas put forward here will be tested in the near future using a simple boundary representation solid modeller recently developed at Cranfield.

REFERENCES

Anderson, C.M. (1983), The New BUILD User's Guide, CAD Group Document 116, Cambridge University Engineering Dept., Cambridge, England.

Faux, I.D. & Pratt, M.J. (1979), Computational Geometry for Design and Manufacture, Ellis Horwood/Halstead Press.

Rossignac, J.R. & Requicha, A.A.G. (1984). Constant Radius Blending in Solid Modeling, CIME July 1984, 65-73.

Tiller, W. (1983). Rational B-splines for Curve and Surface Representation. IEEE CG&A Sept. 1983, 61-69.

Varady, T. & Pratt, M.J. (1985). Design Techniques for the Definition of Solid Objects With Free-Form Surfaces, Computer Aided Geometric Design, to appear.

van Wijk, J.J. (1984), Ray-tracing Objects defined by Sweeping a Sphere, Proc. Eurographics Conf. 1984, Copenhagen, North-Holland Publ. Co.

N85-28570

Boundary to Constructive Solid Geometry Mappings: A Focus on 2-D Issues

Donald P. Peterson
CAD Technology Division

Sandia National Laboratories*
Albuquerque, New Mexico 87185

Most solid modelers (SMs) use either a boundary representation (B-rep) or a constructive solid geometry representation (CSG-rep) as their primary, internal representation. Each method has its advocates; each has its detractors; each has its strong points; each has its drawbacks. The need for both representations is becoming increasingly evident, and it is becoming necessary to be able to convert from one to the other.

The need to go from a CSG-rep to a B-rep (or a simple wireframe rep) has long been apparent because an object that is defined only by a CSG-rep is difficult, if not impossible, for a human to interpret. Before fast, shaded display capabilities became available, the wireframe and the B-rep were the only practicable techniques for psuedo-interactive display. Also, the implementation of the Boolean operations between two solid objects of set union, intersection and difference, which was quickly appreciated as an important capability expected of SMs, utilized techniques of a nature similar to those required in converting from a CSG-rep to a B-rep. As a consequence, the problem of generating a B-rep from a CSG-rep has been extensively studied and is now relatively well understood.

In contrast, until recently, there has been little reason to generate a CSG-rep from a B-rep. However, user experience has now strongly established the convenience of using swept contour (profile) curves in the process of defining objects. Thus, SM vendors have an incentive to provide this input mechanism for their users. But, while the user input data to define a swept object is often trivially related to a B-rep of the object, it has proven more difficult to use these data to construct a CSG-rep of it. Thus, much of the current impetus for looking at B-rep to CSG-rep mappings comes from the developers of CSG-rep SMs who need to provide congenial user interaction. They have pursued various approaches in attacking the problem, but the emphasis has been on finding heuristically justified, algorithmic solutions.

Other applications, e.g., N/C verification, also reinforce the importance of this, and similar, representational mappings.

*This work is performed at Sandia National Labortories, supported by the United States Department of Energy under contract number DE-AC04-76DP0078.

The importance of the problem warrants a better mathematical understanding of it. This talk will primarily address that aspect of it, although some algorithmic considerations will be briefly mentioned.

Although SMs deal with 3-D objects, it turns out that "most" of the geometry for swept objects of current interest is 2-D. Thus, the crux of the matter for generating B-rep to CSG-rep mappings for these cases devolves to solving a 2-D problem. Also, it is more insightful to cast the problem in terms of using "halfspaces" rather than CSG-rep "primitives." This is partly because a solution to the 2-D problem, with 2-D halfspaces, immediately provides a solution to the 3-D problem by simply replacing, in an obvious way, 2-D halfspaces by 3-D halfspaces. As primitives themselves are expressed by (often implicit) binary trees whose leaves are halfspaces, it is true, of course, that a solution with them provides a solution with halfspaces. However, this approach, in general, tends to be profligate in the total number of halfspaces that appears in the CSG-rep of a swept object. Furthermore, as will be shown, the binary tree found for a halfspace solution may be used to determine an excellent binary tree for a solution with primitives.

The problem will be discussed in two phases. The first phase entails finding a CSG-rep that defines the region bounded by a polygonal profile curve. The second phase utilizes the results of the first phase to find a CSG-rep for many non-polygonal profile curves.

A mathematically concise representation of a region bounded by a polygonal will be presented. Namely, any polygonal region bounded by an n sided polygon may be represented by a binary tree which has at most n planar halfspaces as leaves. A structure for this representation and an algorithm for calculating it will be discussed.

If the profile curve is non-polygonal, i.e., has non-straight edges, then a general method for the determination of a concise CSG-rep that defines the region it bounds is not yet known. Nonetheless, while a totally satisfactory solution is lacking, a concise CSG-rep for many regions of interest in CAD applications does exist. This is true, for example, for most regions for which it is possible to determine "an underlying polygonal region" and a set of "edge regions," each associated with a non-straight edge. Most of these regions may be represented by a binary tree that has the polygonal region as one leaf and the edge regions as the other leaves. For such a representation, and for most edges of interest in the present CAD environment, this implies that the region determined by a profile curve with n edges, k of which are non-straight, has a binary tree representation with no more than n+2k halfspace leaves, of which at most n+k halfspaces are distinct. Conditions which assure that this representation exists, and considerations for an algorithm to generate it, will be discussed.

N85-28571

THE DEFINITION AND COMPUTATION OF A METRIC ON PLANE CURVES

THE MEANING OF A "FACE" ON A GEOMETRIC MODEL

James D. Emery
Allied Bendix Aerospace
Kansas City, Mo.

I shall talk about two separate topics as indicated in the title. The first topic concerns the comparison of plane curves. This work arose from the following problem. A surface on a part is generated by a plane curve, which is an interpolating spline. The part is manufactured by an outside vendor using the given interpolation points, but an unknown interpolation technique. The points on the machined curve must be checked to verify that they lie sufficiently close to the original definition. In this case the vendor supplies what is known as the APT-CL-file, which is a set of points defining a piecewise linear curve that defines the actual movement of the machine tool. APT (Automatic Programmed Tool) is a computer program for controlling numerical machine tools. The problem was to calculate a "distance" between these two curves.

There are many other cases where a curve is approximated by some process or algorithm. For example, plotting a curve is usually an approximation technique, and the generation of an offset curve may involve approximation. Other applications include the comparison of surfaces and the inspection of parts. Surfaces may be compared by examining nets of plane curves. Part inspection may involve the determination of a curve or surface from a finite set of measurements.

When the curves are parametric we can use function approximation theory to compare their coordinate functions. However, this would make the comparisons dependent on the particular parameterizations. We could use a canonical parameterization, such as parameterization by arc length, but this is hard to compute and the two curves that we wish to compare may not have the same total arc length.

We shall define a curve to be a locus of points without any underlying parameterization. We shall define a metric on a class of plane curves, give a finite computation of this metric for the case of piecewise linear curves, and show how to approximate curves that have bounded curvature by piecewise linear curves. In this way we can compute a bound on the "distance" between two curves. These techniques have been implemented in Fortran.

We shall prove some standard preliminary propositions. Let (M,d) be a metric space and A a subset of M. Define

$$d(x,A) = \inf\{d(x,y): y \text{ in } A\}$$

DEFINITION. Let A and B be compact sets. The distance between A and B is defined to be

$$d(A,B) = \text{Maximum}\{ \sup_{x\in B} (d(x,A)), \sup_{x\in A} (d(x,B))\}$$

This is the radius of the largest circle that has center on one of the curves, and just touches the other. Note: we have given different meanings to "d." The particular meaning intended is determined by context.

PROPOSITION. Either there exists an x in A so that $d(A,B) = d(x,B)$, or there is an x in B so that $d(A,B) = d(x,A)$.

PROPOSITION. Given a family F of compact sets, (F,d) is a metric space.

We will prove the triangle inequality. Without loss of generality assume $d(A,C) = d(x,C)$ for some x in A. Since $f(w)=d(x,w)$ is continuous on the compact set B, there exists a y in B so that $d(x,B) = d(x,y)$. By definition

$$d(A,B) \geq d(x,B) = d(x,y)$$

Also there exists a z in C so that $d(y,C) = d(y,z)$

We now have

$$d(x,C) \leq d(x,z) \leq d(x,y) + d(y,z)$$

Thus

$$d(A,C) = d(x,C) \leq d(x,B) + d(y,C) \leq d(A,B) + d(B,C)$$

DISTANCE FROM A POINT TO A LINE SEGMENT

This computation is fairly obvious. The distance is either the distance from the point to the line containing the segment, or the distance from the point to one of the end points of the segment.

FINITE COMPUTATION FOR PIECEWISE LINEAR CURVES

The piecewise linear curve is a union of line segments. We call the end points of the segments knots. Given two curves A and B we say that they satisfy the knot condition if the metric is realized at either a knot of A or a knot of B. That is, either there is a knot a in A so that

$$d(A,B) = d(a,B)$$

or there is a knot b in B so that

$$d(A,B) = d(b,A)$$

PROPOSITION. Suppose the pair (A,B) does not satisfy the knot condition. Suppose $d(A,B) = \sup\{d(x,B)\}$. Then there exists a point a in A so that $d(A,B) = d(a,B)$ and the circle

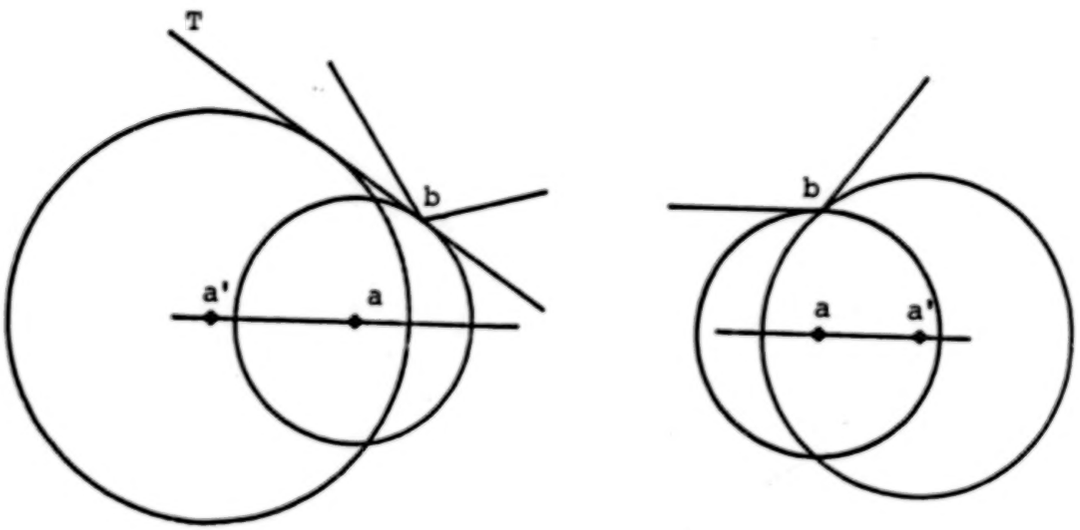
of radius r = d(A,B) and center a meets B in two or more points.

PROOF. Define L(x) to be a line segment that contains point x (there will be two such segments if x is a knot). Assume that for every a such that d(a,B) = d(A,B) the circle of radius r = d(A,B) and center a meets B in only one point. Let a be a point so that

d(a,B) = d(A,B)

Then the circle of radius r = d(A,B) and center a meets B in only one point b and

d(A,B) = d(a,b)



Let T be the line tangent to the circle at b. There are three cases.

(1) L(a) and T are not parallel.

(2) L(a) is parallel to T and some L(b) is not parallel to T.

(3) L(a) and every L(b) are parallel to T.

Case 1. The line T separates the plane into two half spaces. The circle lies in one and every L(b) lies in the other. This follows from the definition of d(A,B). Because the circle touches B only at b and T is not parallel to L(a), we may move the center a to a new center a' so that the radius of the circle increases, and does not contain a point of B in its interior (see the left figure). But this contradicts the definition of d(A,B).

Case 2. T is parallel to L(a) and some L(b) does not lie in T. Again we may slightly alter a to a' to increase the

radius of the circle (see the right figure). This is a contradiction.

Case 3. L(a) is parallel to T and every L(b) lies in T. Then we may translate the circle along L(a) until one of the following occurs.

(i) The center becomes a knot of A.

(ii) The circle contacts a second point of B.

(iii) Some L(b) does not lie in T.

Event (i) cannot happen because our curves do not satisfy the knot condition, and (ii) cannot happen because our assumption was that a could not be chosen so that the circle contacts B in two or more points. Therefore (iii) must occur. However, we have shown that cases (1) and (2) above are contradictions. Thus (iii) is a contradiction. Therefore our original assumption is false and the proposition is proved.

With the aid of this proposition we can construct an algorithm to compute d(A,B). Obviously the circle can not contact a single segment in two points. Thus we need only compute all circles that have centers on A and just touch B at two points. The points where the circle contacts B must be either tangent points, or knots. That is if the circle C(a,r), which has center a and radius r, contacts B at b where b is an element of segment I, and b is not an end point, then C(a,r) is tangent to I. There are only three cases.

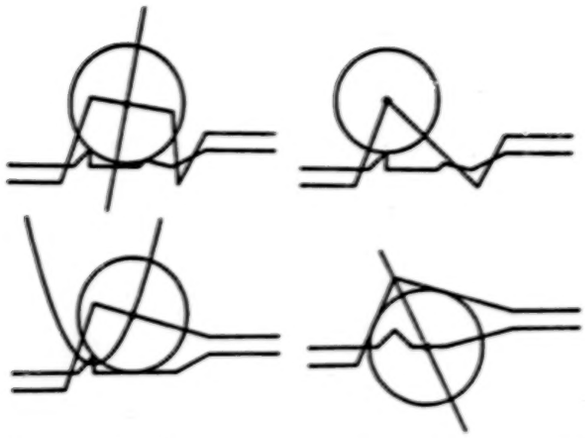
(1) The center a lies on the locus of points equidistant from two knots of B.

(2) The center lies on the locus of points equidistant from two lines. The lines contain segments of B.

(3) The center lies on a locus of points equidistant from a knot of B and a line, which contains a segment of B.

Thus we need only compute perpendicular bisectors of segments, angle bisectors of angles, and parabolas. The parabolas are defined by the knot-line pairs of B. The knot is the focus and the line is the directrix.

To find all the candidates for centers, we intersect the various loci with the curve A. We must also of course reverse the roles of A and B in case d(A,B) = d(b,A) for some b in B. There are then four cases in total, including the case of a center occurring at a knot. These cases are illustrated below.



The various loci are computed using projective representations and being careful to avoid excessive round-off error and making sure that special cases are handled correctly.

We can avoid doing the computations for all point-point, point-line, and line-line pairs by computing certain distances between segments and establishing certain inequalities.

THE PIECEWISE LINEAR APPROXIMATION

We will show that if we know a bound on the curvature then we can construct a piecewise linear curve that approximates a given curve arbitrarily closely.

Let g_r be a function whose graph is a circular arc of radius r. The center of the arc is below the x-axis and it passes through the points (a,0) and (b,0) where a < b. Let $\kappa(f)$ be the magnitude of the curvature of f.

PROPOSITION. Let f be a twice differentiable function such that f(a) = f(b) = 0. If f is greater than g_r anywhere on the interval (a,b), then there is a point in (a,b) where

$$\kappa(f) > 1/r.$$

PROOF. Let $h = f - g_r$. Suppose that f is greater than g_r at some point of (a,b). Let x be a point of (a,b) where the maximum value of h is attained. Therefore

$$h'(x) = 0 \quad \text{and} \quad h''(x) \leq 0$$

Now $g_r'' < 0$, so

$$h''(x) = f''(x) + \text{Abs}(g_r''(x)) \leq 0$$

Therefore

$$\text{Abs}(f''(x)) \geq \text{Abs}(g_r''(x))$$

$f'(x) = g_r'(x)$, because $h'(x) = 0$. It follows from the expression for the curvature that

$\kappa(f) \geq \kappa(g_r) = 1/r.$

We can repeat the argument with a slightly smaller r. Thus

$\kappa(f) > 1/r.$

COROLLARY. If $\kappa(f) \leq 1/r$ on (a,b) then $f \leq g_r$ on (a,b).

PROPOSITION. Let C be a curve connecting two points. Suppose $\kappa(C) \leq 1/r$. Let L be the line segment joining the two points. Suppose any line passing through the segment and perpendicular to it meets C in exactly one point. Let the length of the segment be ℓ. If

$\ell/2 < r$

then

$d(C,L) \leq r - \text{Sqrt}(r^2 - \ell^2/4)$

This proposition provides an obvious construction of a piecewise linear curve that approximates a given curve to arbitrary accuracy provided we know a bound on the curvature.

Perhaps these techniques can be generalized to surfaces using approximating polygons. However, for curves the piecewise linear approximation in some sense approximates both curvature and arc length. For surfaces the analog is not true. It is well known that triangulations can be found for surfaces in such a manner that the triangles approximate the surface arbitrarily closely, yet have arbitrarily large surface area (see [1]).

FACES

The second topic I wish to talk about concerns faces on geometric models. Systems for representing the surface of a solid model have been constructed (see [4] for example). These systems give both a topological, that is an abstract, and a geometric representation. The concepts involved are such things as vertices, edges, loops, faces, and surfaces. The topological representation leads to an obvious triangulation and thus to a simplicial complex. Therefore the model may be treated conventionally. This simplicial complex has a geometric realization. There is a dual representation involving faces on specific surfaces. These faces are bounded by "loops." The questions to be discussed are, "Under what circumstances do these geometrical faces make sense?" and "How can they be explicitly defined?" Also, "When are these 'geometrical' faces homeomorphic to the realization of the abstract (topological) face?"

Faces here are essentially discs with holes. Algebraic topology in effect treats faces as discs with no holes. Storage and computational constraints force us to deal with this new type of face.

REFERENCES

1. Smith, Kennen, Primer of Modern Analysis, Bogden and Quigley, New York, 1971.

2. Agoston, Max, Algebraic Topology, Marcel Dekker, New York, 1976.

3. Giblin, P. J., Graphs, Surfaces and Homology, Chapman and Hall, 1977.

4. Geometric Modeling Project Boundary File Design (XBF-2), R-81-GM-02.1, Computer Aided Manufacturing International, Arlington, Texas, 1981.

N85-28572

SOME NEGATIVE RESULTS IN N-SIDED PATCHES

Malcolmb Sabin
Finite Element Graphics Systems, Limited
Cambridge, UK

To ease surface definition in situations of complex topology, patches with other than four sides and with guaranteed continuity of at least slope with adjacent patches are needed.

There are known techniques in the literature for three- and five-sided patches of reasonable economy, and known transfinite methods for many sides.

The author has looked for a suitable six-sided patch for nearly 20 years. After reviewing the methods in the literature, he will identify some of the literature. He will show some of the conditions that such a patch must satisfy and how certain simpler approaches are not effective.

PRECEDING PAGE BLANK NOT FILMED

N85-28573

EXAMINATION OF THE CIRCLE SPLINE ROUTINE

Ronald M. Dolin and Dwight L. Jaeger

Los Alamos National Laboratory
Albuquerque, New Mexico

ABSTRACT

The Circle Spline routine was developed by D. L. Jaeger and is currently being used at the Los Alamos National Laboratory for generating both two- and three-dimensional spline curves. It was developed for use in ESCHER, a mesh generating routine written by W. R. Oakes, to provide a computationally simple and efficient method for building meshes along curved surfaces. Circle Spline is a parametric linear blending spline. Because many computerized machining operations involve circular shapes, the Circle Spline is well suited for both the design and manufacturing processes and shows promise as an alternative to the spline methods currently supported by the Initial Graphics Exchange Specification (IGES).

Circle Spline constructs a spline by generating a series of circular arcs, each of which passes through three successive data points (Pi-1, Pi, Pi+1), where i = 2,...,n and n = the number of data points. Thus, for n data points, n-2 arcs are constructed. These segmented arcs are then blended together, using a linear blending function, to represent the desired curve. Circle Splines need only have the data point coordinates specified. Because it is both a two- and three-dimensional routine, the three coordinates must always be specified, even for two-dimensional applications, in which case the third coordinate is set to zero. The spline passes through all the data points and these points do not need to be evenly spaced. Circle Spline does not require first or second derivative continuity, but instead has a different approach for determining aesthetic qualities such as smoothness and fairness.

The routine generates the spline by taking three successive data points (Pi-1, Pi, Pi+1), calculating the parametric location of a circle center $\underline{R}$, which passes through the points, and constructing an arc, of radius $\underline{R}$, through these points. With this information, the curved arc length between data points can be calculated. The value of the curved arc length for each interval is then stored. The routine then increments along the curve one data point and repeats the process for data points (Pi, Pi+1, Pi+2). With the exception of the two end point pairs (Pi, Pi+1) and (Pn-1, Pn), each pair of points (Pi, Pi+1) i = 2,...,n-2, will have two arcs passing through it. This means that each interval (i, i+1), i = 2,...n-2, has two values for the curved arc length (one for each arc that passes through it). The routine determines which of these two curved arc lengths is larger and retains that value.

Once all the arc lengths are calculated, the total arc length and parametric position of each data point are calculated. The routine uses a rational parameterization. If the sum of the curved arc lengths is S, the parametric value of each data point becomes the ratio of the spline length from the start of the curve to the data point divided by S. The parameter "u" will then range from $0 \leq u \leq 1$.

PRECEDING PAGE BLANK NOT FILMED

The use of the larger curved arc length within each interval was based on Circle Splines implementation in ESCHER. Better mesh point spacing is achieved using this scheme. A better parameterization may be achieved by using a function of the two arc lengths. In any event, using the curved arc lengths to calculate the total spline length no doubt leads to a better parameterization than the chordal arc length method used by most spline routines.

The two arc segments passing through each pair of data points are blended into a single curve. The user specifies the number of subpoints that will be used for the blending operation. These subpoints are evenly spaced parametrically along the curve. The code can also be used so that the subpoints are weighted, e.g., left, right, or center weighted. An automatic scheme whereby the subpoints are clustered in areas of varying curvature may produce a smoother blend.

The data points and the subpoints may be coalescent at points along the curve without affecting the spline. Because the data points are allowed to be unevenly spaced, the number of subpoints that fall within a data point interval may vary. The number of subpoints within an interval represents the number available for a particular blending operation.

Let arc-i represent the arc passing through points (Pi-1, Pi, Pi+1), and arc-i+1 represent the arc passing through points (Pi, Pi+1, Pi+2). Both arcs will pass through points Pi and Pi+1. For the purpose of example, suppose 10 subpoints lie within this interval. These 10 points will be evenly spaced along the interval. The routine will determine how far each subpoint is from points Pi and Pi+1 and based on this information will calculate the subpoints position relative to the two arcs. The closer a subpoint is to Pi, the closer it will be positioned to arc-i. As the subpoint placement approaches Pi+1, their positions will move closer to arc-i+1. At the start of an interval the blended curve is tangent to arc-i, at the end of the interval it is tangent to arc-i+1, in the center of the interval each arc will have an equal influence on the subpoint positioning, and the position of the blended spline will be in the geometric center of the plane defined by the two arcs.

If Pb equals the parametric value of a subpoint within the interval $Pi \leq Pb \leq Pi+1$, its position will be determined using the following equations

$$Ratio = (Pb - Pi) / (Pi+1 - Pi)$$

$$Ratm1 = 1 - Ratio$$

The value of the X-coordinate is thus,

$$XX(i) = Ratm1 * X(i) + Ratio * X(i+1)$$

where X(i) is the X-coordinate of Pi and X(i+1) is the X-coordinate of Pi+1. YY(i) and ZZ(i) arc calculated similarly. Using this scheme intuitively suggests that the blended curve can have, at most, one inflection point per interval.

Spline smoothness can be measured by examining how well the circle centers for each arc relate. Because curvature is the inverse of the radius, when the curvature is zero, $\underline{R}$ will approach infinity. Thus, the $\underline{R}$s provide an excellent way of monitoring the curvature. Many manufactured items contain circular shapes, for

example, holes, rounded edges, etc.; thus, the Circle Spline is well suited for Computer Aided Manufacturing (CAM).

The Circle Spline is computationally efficient and easy to understand. It was recently compared to the Wilson-Fowler spline in two dimensions; it compared favorably for most applications, but as might be expected, pathological cases existed where each routine produced better results. Circle Spline can be used in both two and three dimensions and its applicability to the design and manufacturing process makes its future use promising.

N85-28574

Convex Interpolating Splines of Arbitrary Degree

Abstract

Edward Neuman*
Department of Mathematics
Southern Illinois University
Carbondale, Illinois 62901

1. In curve fitting problems that arise in science or engineering we often demand that our approximating function be shape preserving in the sense that the approximation is convex when the data are convex or that the approximation is monotone and convex when the data are monotone and convex. In this paper we construct such shape preserving approximations by interpolating the data with polynomial splines of arbitrary degree. We formulate a regularity condition on the data which insures the existence of such a shape preserving spline, we present an algorithm for its construction, and we bound the uniform norm of the error which results when the algorithm is used to produce an approximation to a given $f \in C[a,b]$.

2. Let $\Delta_n : a = x_0 < x_1 < \ldots < x_n = b$ denote an arbitrary but fixed partition of the interval $[a,b]$ with knots x_i, let $h_i = x_{i+1} - x_i$, $i = 0,1,\ldots,n-1$, and let $h = \max h_i$. Let $Sp(k, \ell, \Delta_n)$ denote the space of polynomial splines of degree k and deficiency $k - \ell$ and assume that $k = 3,4,\ldots$ and $\ell = 1,\ldots, [(k - 1)/2]$.

---

*On leave from the Institute of Computer Science, University of Wroclaw, Poland.

PRECEDING PAGE BLANK NOT FILMED

Given real data (x_0,f_0), ..., (x_n,f_n) we seek $s \in Sp(k, \ell, \Delta_n)$ such that

(P) $\quad s(x_i) = f_i, \quad i = 0,1,\ldots,n$ and s is convex on [a,b].

The existence of a solution to (P) depends on the values of

$$\sigma_i = (f_{i+1} - f_i)/h_i, \quad i = 0,1,\ldots,n-1.$$

To construct s we set

(1) $\quad s(x) = s_i(x), \quad x_i \le x \le x_{i+1}, \quad i = 0,1,\ldots,n-1$

where

(2) $$s_i(x) = f_i\phi_0(t) + f_{i+1}\phi_1(t) + h_i[p_i\Psi_0(t) - p_{i+1}\Psi_1(t)],$$
$$t = (x - x_i)/h_i,$$

where $p_0,p_1,\ldots,p_n$ are parameters to be determined later and

$$\phi_0(t) = \int_t^1 M_k^m(s)ds, \qquad \phi_1(t) = \int_0^t M_k^m(s)ds,$$

$$\Psi_0(t) = t\int_t^1 M_{k-1}^m(s)ds, \qquad \Psi_1(t) = (1 - t)\int_0^t M_{k-1}^{m-1}(s)ds,$$

where M_i^j denote the B-spline of degree i - 1 with knots $0 = t_0 = \ldots = t_j < t_{j+1} = \ldots = t_i = 1$. Clearly $s \in Sp(k, \ell, \Delta_n)$ with $\ell = \min\{k-m-1, m\} \le [(k - 1)/2]$.

Theorem 1. The spline function s given by (1) - (2) solves the problem (P) if and only if

(3) $$\frac{(k - m)p_i + m\, p_{i+1}}{k} \le \sigma_i \le \frac{(k-m-1)p_i + (m + 1)p_{i+1}}{k}$$

for $i = 0,1,\ldots,n-1$. If in addition $p_0 \ge 0$, then s is also nondecreasing on [a,b].

3. The parameters p_i which appear in (3) can be constructed as follows. We initially define

$$\ell_i = \frac{k\sigma_i - (m+1)\sigma_{i+1}}{k-m-1}, \quad i = 0,\ldots,n-2, \quad \ell_{n-1} = \ell_n = -\infty,$$

$$u_0 = -\infty, \quad u_i = \frac{k\sigma_{i-1} - (k-m-1)p_{i-1}}{m+1}, \quad i = 1,\ldots,n,$$

$$v_0 = \infty, \quad v_i = \frac{k\sigma_{i-1} - (k-m)p_{i-1}}{m}, \quad i = 1,\ldots,n,$$

and then for each $i = 0,1,\ldots,n$ set $c_i = \max\{u_i, \ell_i\}$, $d_i = \min\{v_i, \sigma_i\}$.

Theorem 2. Let $\sigma_1 \geq \sigma_0$ and let $\sigma_n = +\infty$. If

$$(m+1)(\sigma_{i+1} - \sigma_i) \geq (k-m-1)(\sigma_i - \sigma_{i-1}), \quad i = 1,\ldots,n-2,$$

then $c_i \leq d_i$, $i = 0,\ldots,n$ and any choice of $p_i \in [c_i, d_i]$, $i = 0,\ldots,n$ makes (1) - (2) a solution of (P).

4. The error which results when the above procedure is used to construct a polynomial spline approximation to a continuous function f may be bounded as follows:

Theorem 3. Let $f \in C[a,b]$ be nondecreasing and convex on $[a,b]$, and let s be the spline interpolant produced by using (1) - (3) with $f_i = f(x_i)$, $i = 0,\ldots,n$ and with $p_0 \geq 0$. Then

$$\|f - s\|_\infty \leq \gamma_{k,m}\omega(f;h)$$

where $\omega(f;\cdot)$ is the modulus of continuity of f and

$$\gamma_{k,m} = 1 + A\cdot k/m \quad \text{and} \quad \frac{1}{2} - \binom{k-1}{m}2^{-k} \leq A \leq 1.$$

Moreover, for $k = 2q-1$ and $m = q-1$ $(q > 1)$

$$\frac{7}{4} \leq \gamma_{3,1} < \gamma_{5,2} < \ldots \leq 2.$$

N85-28575

A NATURAL BIAS APPROACH TO CARDINAL SPLINE CURVES*

G. Yates Fletcher and David F. McAllister
North Carolina State University
Raleigh, North Carolina

The cardinal spline approach to defining interpolatory curves has been of recent interest in the application of computer graphics to modelling and animation problems [1], [2]. In our formulation of this interpolation problem (C) we desire to obtain a vector valued function V which forms a C^1 map from [0,n] into the plane in such a way that V(i) and V'(i) attain prespecified (vector) values P_i and T_i for the integers i=0, 1,...n. The intent is that the planar curves V(s) pass through each point P_i in the direction corresponding to T_i. In actual practice it is common for only the interpolation points P_i to be user specified. Suitable tangents T_i are then provided by some heuristic to complete the specification of (C). If, for instance, the T_i values are computed as

$$T_i = (P_{i+1} - P_{i-1})/2 \qquad (1)$$

(each tangent is the average of the incoming and outgoing chord vectors to the adjacent points), then the solution to (C) is the Catmull-Rom spline. Once the tangents T_i have been defined we may write the cardinal spline curve V(s) as a collection of segments

$$V(s) = V_i(s-i) \qquad i \leq s \leq i+1,\ i=1,\ldots,n-1$$

where each curve segment V_i is a Hermite interpolant which may be written simply in terms of the cardinal Hermite basis. Defining a vector function of a real variable s by

$$H(s) = [1-3s^2+2s^3,\ 3s^2-2s^3,\ s-2s^2+2^3,\ -s^2+s^3]$$

we may write

$$V_i(s) = H(s)*[P_i,\ P_{i+1},\ T_i,\ T_{i+1}]^t. \quad 0\leq s\leq 1 \qquad (2)$$

For most applications the magnitudes of the tangents T_i can be allowed to vary without effecting their primary function which is to furnish a suitable direction for the interpolating curve at each node. The effects of these variations on the resulting curve are perceived subjectively as "shape" changes. They do not alter the fundamental property of such curves - that they possess a continuously varying unit tangent vector. The effects are best understood by considering a single section of the curve which passes through points P_0 and P_1 with tangents T_0 and T_1. In general, increasing the magnitude of a tangent vector forces the interpolating curve to "stay closer longer" to the line on which the tangent vector lies. Thus increasing the magnitude of one tangent at the expense of the other will "pull" the curve in its direction. This effect is called "bias." Increasing the magnitudes of both tangents proportion-

*Research supported in part by NASA Grant NAF-103-S4.

PRECEDING PAGE BLANK NOT FILMED

ally, forces the bend more toward the middle; decreasing the magnitudes allows the curve to "leave the tangent directions sooner" and causes a flattening effect called "tension." A discussion of these effects is given in [3].

The important issue here is the point of view that the salient feature of the tangent vectors is the "direction" they impart to the curve and that their magnitudes may be adjusted to effect shape control. A natural question to ask in this context is the following - given the two endpoints and corresponding tangent directions which define a segment on the curve, what should be the default bias, i.e. the ratio of their magnitudes which serves as a base for further adjustments? We feel that this question has a natural answer based on the geometry of each curve segment, one that yields curves with interesting and desirable shape properties. We give a brief summary of our treatment.

Given a segment with endpoints P_0 and P_1 and tangents T_0 and T_1 we solve the equation

$$xT_0 + yT_1 = P_1 - P_0 \qquad (3)$$

for scalar values x and y. There are three primary cases to consider:

i) $xy > 0$
ii) $xy < 0$
iii) (3) has no solution

Other cases occur when T_0 and T_1 lie on the line containing P_0 and P_1 and are degeneracies which we discuss separately. For case i) we rescale T_0 and T_1 as xT_0 and yT_1 respectively. For cases ii) or iii) we solve (3) with T_0 replaced by T_0', its "reflection" in the line containing P_0 and P_1. The x and y thus obtained will now satisfy $xy > 0$, and we proceed as before rescaling T_0 (the original T_0) as xT_0 and T_1 as yT_1. This rescaling is the means by which a "natural" bias for the segment is obtained. See figure 1.

When the new tangents are multiplied by a common "tension" factor t, the resulting curve segment defined by (2) has the following properties which form our justification for using the word "natural" to describe the biasing technique.

For case i)

- For $t = 3$ the segment has zero curvature at both end points P_0 and P_1. For $t < 3$ the segment will have no inflection points (changes of concavity). For $t > 3$ the segment will have exactly two inflection points.

- For $t = 2$ the 3rd order terms vanish and the resulting curve is in fact a parametric quadratic.

- For $t = 6$ the segment will have a cusp where the curve parameter $s = .5$.

For cases ii) and iii)

- The curve will intersect the line segment S connecting P_0 and P_1 exactly once (let Q denote this point).

- The maximum deviations of the curve from S on either side of Q will be the same.

- For t = 3 the tangent to the curve at Q is perpendicular to S.

As a consequence of our results for segments falling under case i) we can prove the following theorem.

Let $P_1, P_2, \ldots P_n$ be vertices of a convex planar polygon P given in order of adjacency, and let tangents T_i be defined by (1) with

$$T_0 = T_{n-1} \quad \text{and} \quad T_{n+1} = T_1$$

The closed cardinal spline curve (whose segments are defined by (2) where the tangents are biased by the procedure we have outlined and controlled by a tension factor t) will form the boundary of a convex set C containing P for any value of t such that $0 < t < 3$.

In short, a convex control polygon will yield a convex interpolant. We can also give examples where the interpolant fails to be convex if the tangents are left unbiased.

References

1. Alvy Ray Smith, "Spline Tutorial Notes - Technical Memo No. 77", SIGGRAPH '83 Tutorial Notes: Introduction to Computer Animation, pp. 64-75, July, 1983.

2. D. H. Kochanek, and R. H. Bartels, "Interpolating Splines with Local Tension, Continuity, and Bias Control", SIGGRAPH '84, Minneapolis, July 23-27, 1984, pp. 33-41.

3. J. D. Foley, and A. Van Dam, Fundamentals of Interactive Computer Graphics, pp. 514-523, Addison-Wesley, 1983.

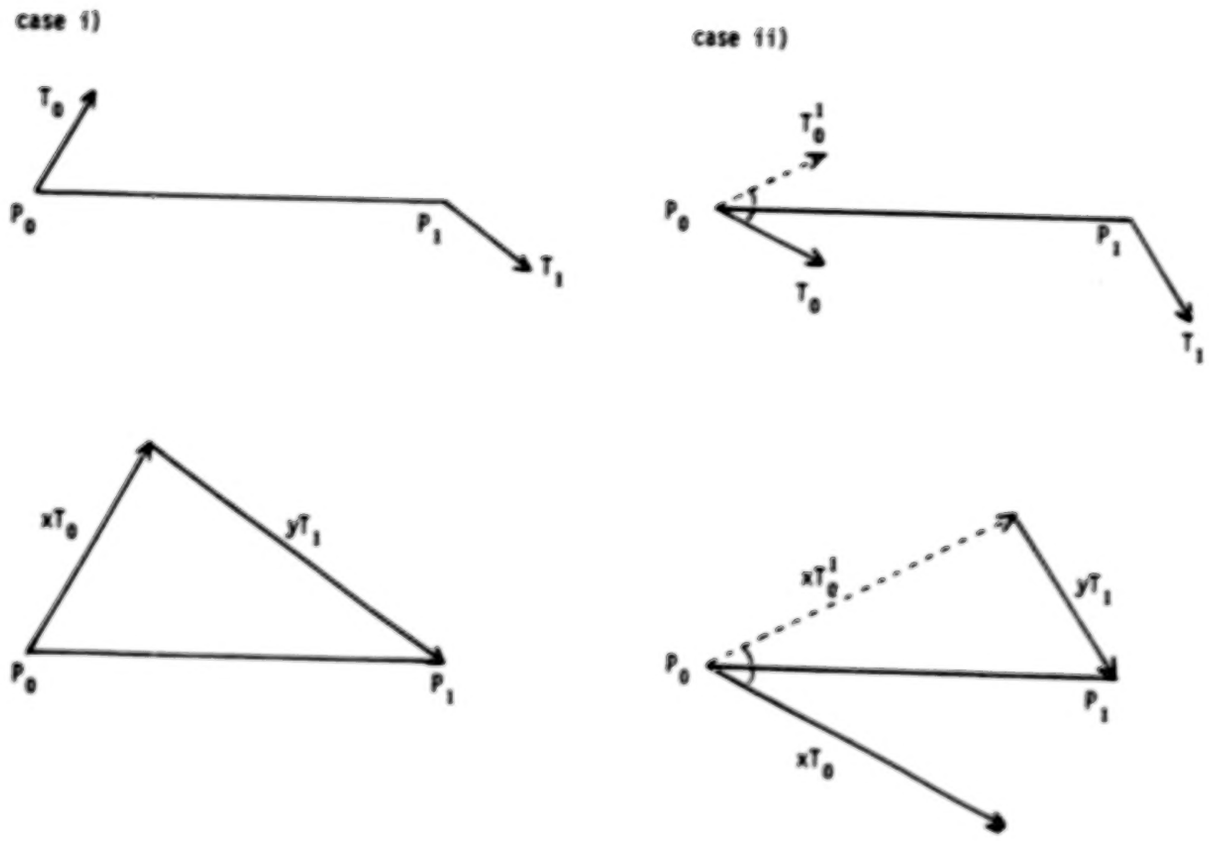

case i)
T_0
P_0
P_1
T_1
xT_0
yT_1
P_0
P_1
case ii)
T_0^1
P_0
P_1
T_0
T_1
xT_0^1
yT_1
P_0
P_1
xT_0

Figure 1

N85-28576

Adapting Shape Parameters for Cubic Bézier Curves

David Isacoff
and
Michael J. Bailey
CADLAB, Purdue University
134 Potter Engineering Center
West Lafayette, IN 47907

Bézier curves are a popular and well established tool in Computer Aided Geometric Design. One of the major drawbacks of the Bézier method, however, is that the curves often bear little resemblance to their control polygons. As a result, it becomes increasingly difficult to obtain anything but a rough outline of the desired shape. One possible solution would be to manipulate the curve itself instead of the control polygon. The following paper introduces into the standard cubic Bézier curve form two shape parameters, γ_1 and γ_2. These parameters give the user the ability to manipulate the curve while the control polygon retains its original form, thereby providing a more intuitive feel for the necessary changes to the curve in order to achieve the desired shape.

A 4^{th} order (3^{rd} degree) Bézier curve [BEZI72] is defined as

$$C(t) = \sum_{i=0}^{3} p_i \phi_i(t) \qquad [1]$$

for $0 \leq t \leq 1$
where

$$\phi_i(t) = \binom{3}{i} t^i (1-t)^{3-i}$$

and the p_i's are vertices of the Bézier control polygon. Expanding [1] gives:

$$C(t) = \phi_0(t)p_0 + \phi_1(t)p_1 + \phi_2(t)p_2 + \phi_3(t)p_3 \qquad [2]$$

In matrix form, [CLAR81] the cubic Bézier becomes:

$$C(t) = [t^3\ t^2\ t\ 1]\begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}\begin{bmatrix} p_0 \\ p_1 \\ p_2 \\ p_3 \end{bmatrix} \qquad [3]$$

$$= \mathbf{T}\,\mathbf{M}\,\mathbf{P}$$

where

T is the primitive polynomial basis

M is the basis change matrix and

P is the geometry vector

Two important properties of the Bézier curve are

$$\sum_i \phi_i(t) = 1 \qquad 0 \le t \le 1 \qquad [4]$$

$$\phi_i(t) > 0 \qquad 0 \le t \le 1 \qquad [5]$$

The first property is necessary for translational and rotational invariance. This means the curve will be independent of the choice of the coordinate system. In addition to the first property, the second property is necessary for the curve to lie within the convex hull generated by the vertices of its control polygon.

Evaluating C(t) at its endpoints (t = 0, t = 1) we get

$$C(0) = p_0 \qquad [6]$$

$$C(1) = p_3$$

In a similar manner when we evaluate $C'(t)$ at its endpoints (t = 0, t = 1) we get

$$C'(0) = 3(p_1 - p_0) \qquad [7]$$

$$C'(1) = 3(p_3 - p_2)$$

By examining $C'(0)$ and $C'(1)$ we can see that the cubic Bézier representation expresses the tangents at the endpoints in terms of difference vectors between two geometric points multiplied by a constant. Therefore equation [7] can be rewritten as

$$C'(0) = \gamma(p_1 - p_0) \qquad [8]$$

$$C'(1) = \gamma(p_3 - p_2)$$

where $\gamma = 3$ in the case of the cubic Bézier curve.

Now suppose we allow a different γ for $C'(0)$ and $C'(1)$. For example

$$C'(0) = \gamma_1(p_1 - p_0) \qquad [9]$$

$$C'(1) = \gamma_2(p_3 - p_2)$$

How does this new change affect our original C(t)? What are the new $\phi_i(t)$ needed to maintain the two previously stated properties?

The effect on C(t) can be observed by looking at the new basis change matrix M_γ. Substituting into equation [3] we get

$$C(t) = T\, M_\gamma\, P$$

where

$$M_\gamma = \begin{bmatrix} 2-\gamma_1 & \gamma_1 & -\gamma_2 & \gamma_2-2 \\ 2\gamma_1-3 & -2\gamma_1 & \gamma_2 & 3-\gamma_2 \\ -\gamma_1 & \gamma_1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

Expanding yields

$$C(t) = \psi_0(t)p_0 + \psi_1(t)p_1 + \psi_2(t)p_2 + \psi_3(t)p_3$$

where

$$\psi_0(t) = (1-t)^2(1 + 2t - \gamma_1 t)$$

$$\psi_1(t) = \gamma_1(t-1)^2 t$$

$$\psi_2(t) = \gamma_2(1-t)t^2$$

$$\psi_3(t) = t^2(\gamma_2 t - 2t - \gamma_2 + 3)$$

Note that $\sum_i \psi_i(t) = 1 \qquad 0 \le t \le 1$

and $\psi_i(t) > 0 \qquad 0 \le t \le 1$, if $0 < \gamma_1, \gamma_2 \le 3$

The restrictions of γ_1 and γ_2 are necessary in order to maintain the convex hull property. If $\gamma_1, \gamma_2 > 3$ convexity may still be maintained, but is not guaranteed as it is with the standard Bézier curves (Figure 1). The reason the convex hull property is desirable is that it gives the designer an initial feel for the shape of the curve. Once the rough outline has been drawn, however, this criterion may be temporarily sacrificed. In Figure 2 we can see the rough outline of an airfoil. Figure 3 demonstrates a typical refinement process and Figure 4 shows the completed design. After the final design is reached, or at any intermediate step, it is possible to generate the new Bézier control polygon for the existing curve, thus producing a natural method of step-wise refinement.

This work was supported by Control Data Corporation under grant 81P04.

REFERENCES

[BEZI72] Bézier, Pierre. *Numerical Control: Mathematics and Applications*, John Wiley & Sons, New York, 1970.

[CLAR81] Clark, James H. "Parametric Curves, Surfaces and Volumes in Computer Graphics and Computer-Aided Geometric Design," Technical Report 221, Computer Systems Laboratory, Stanford University, Palo Alto, California, November 1981.

[KUET76] *Foundations of Aerodynamics: Bases of Aerodynamic Design*, John Wiley & Sons, New York, 1976.

[PIEG84] Piegl, L. "A Generalization of the Bernstein-Bézier Method," *Computer-Aided Design*, Vol 16, No 4, July 1984. pp 209-215.

[TIMM80] Timmer, Henry G. "Alternative Representation for Parametric Cubic Curves and Surfaces," *Computer-Aided Design*, Vol 12, No 1, January 1980. pp 25-28.

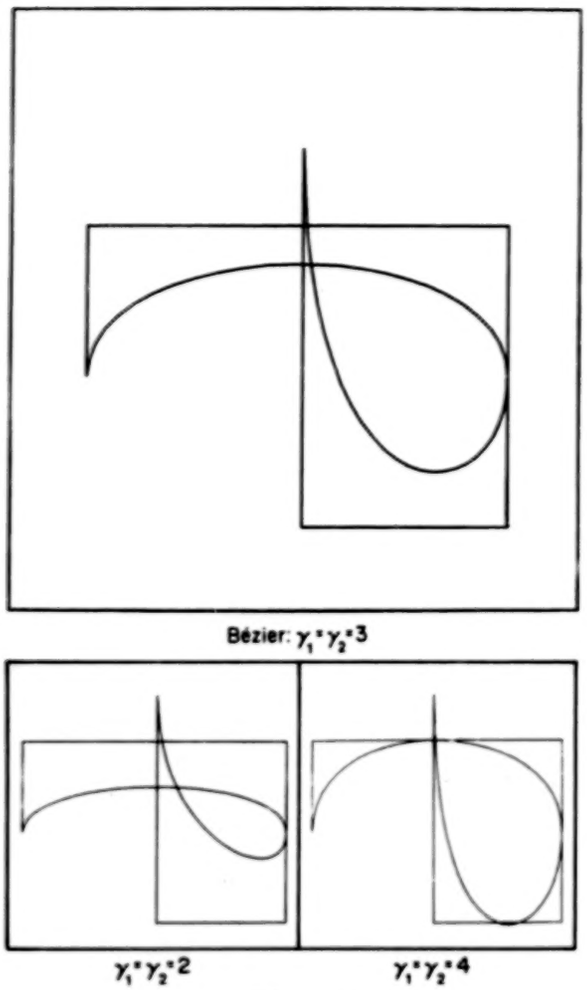


Figure 1

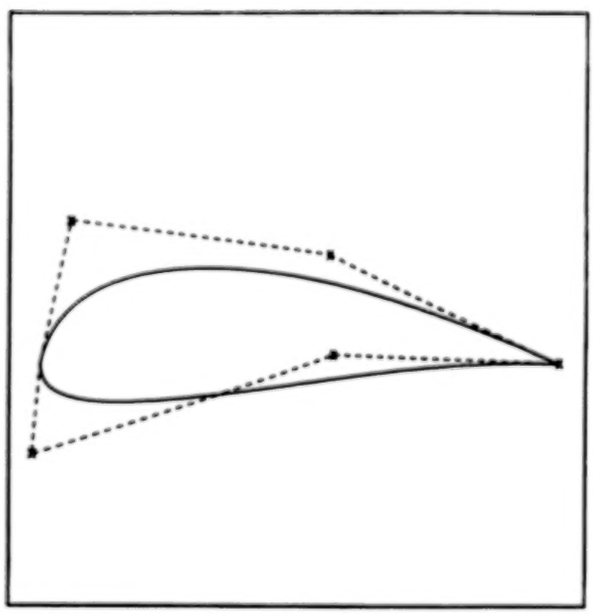


Figure 2: Rough Outline Using Standard Bézier Curves

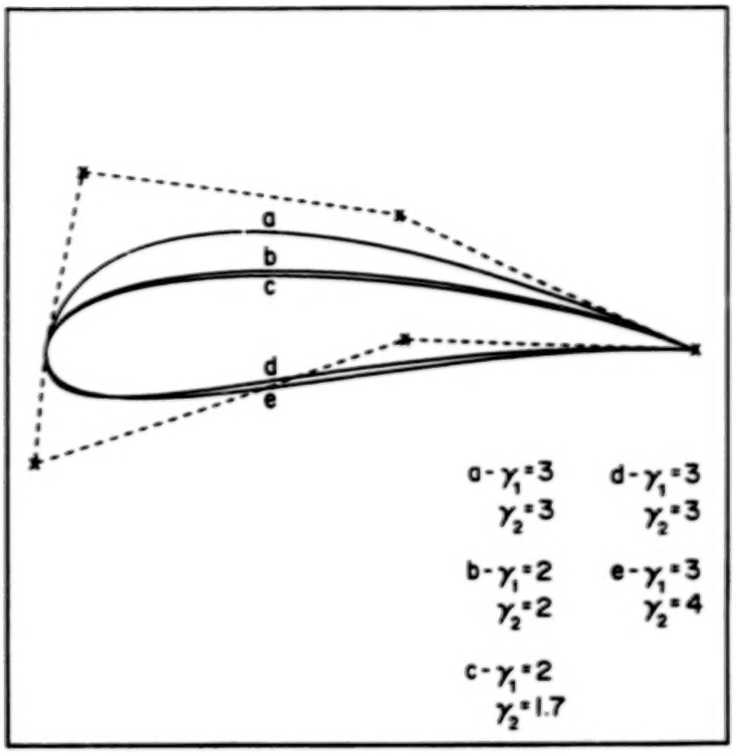


Figure 3: Refinement Process

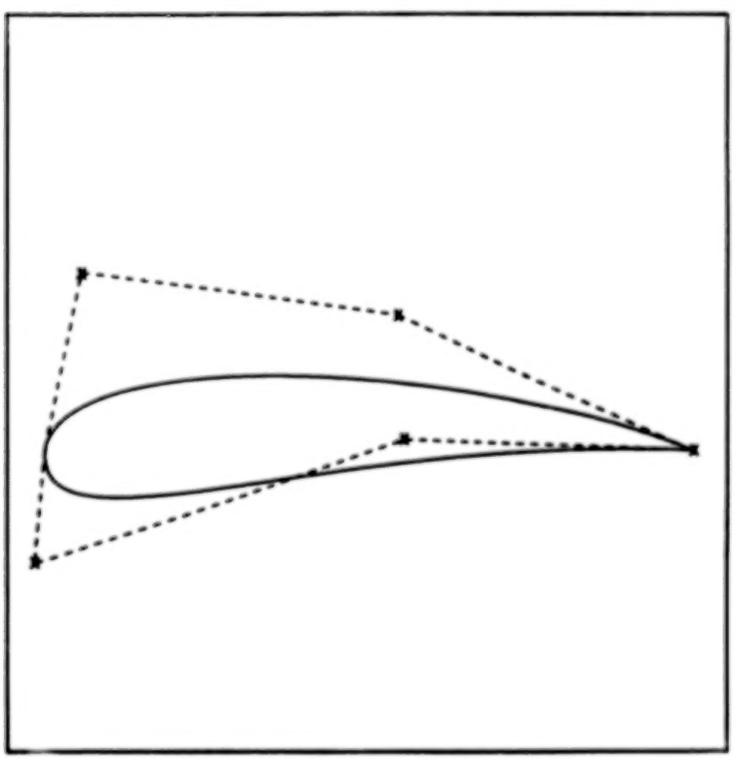


Figure 4: Completed Design with Original Control Polygon

N85-28577

Triangular Bernstein-Bézier Patches, A Survey and New Results

Gerald Farin
Department of Mathematics
University of Utah
Salt Lake City, UT 84112

Triangular Bernstein-Bézier patches are an alternative to the standard rectangular ones. They are maps of triangular domains into R^3, where the patches are described by control nets similar to the rectangular case. One difference is the definition of surface normals at the patch corners: rectangular patches have inherent problems with so-called degenerate corners, i.e. edges collapsing into one point. Triangular patches do not have these problems. Since triangular shapes arise rather frequently in the design of complicated surfaces (e.g. interior car body panels), it seems that for such surfaces triangular patches are a promising addition to the usually employed rectangular patches.

Another major application is the formulation of interpolants to solve the scattered data problem: here Bernstein-Bézier patches provide a very powerful tool; they can be used to discover new schemes as well as to discover new properties of existing methods.

A survey of known results is given, including

Subdivision: The recursive evaluation of triangular Bernstein-Bézier patches generates control nets for the three subtriangles of the original patch that are generated by the point of evaluation. This algorithm is important for applications such as contouring and rendering: since every patch is contained in the convex hull of its control net, a simple rejection procedure exists. If a patch cannot be rejected, it will be subdivided and the rejection test will be performed for all subtriangles.

Continuity conditions: For complicated surfaces, continuity of adjacent patches must be ensured. Conditions for continuity of arbitrary order are derived from the recursive evaluation procedure. In the design of surfaces, simple continuity is sometimes too restrictive and the concept of "visual continuity" must be invoked. This means that along common patch boundaries, tangent planes from both patches must coincide, but partials or directional derivatives need not be coplanar. Smoothness conditions of this form are easily formulated in terms of control nets.

Connection to rectangular patches: For hybrid surfaces, composed of both rectangular and triangular patches, continuity conditions must be established. In most cases this again means "visual continuity". The resulting conditions are similar to the ones for triangle/triangle conditions.

Interpolants in Bernstein-Bézier form: Triangular interpolants, such as the Clough-Tocher element,[1] are presented in Bernstein-Bézier form. These interpolants aim at the solution of the scattered data problem and define a surface that is piecewise defined over a triangulation of the given data points. The surface will be a Bernstein-Bézier patch over each triangle and will satisfy certain imposed continuity conditions. The Bernstein-Bézier method turns out to be a very useful tool in analyzing such interpolants as well as facilitating their development.

Several new results are presented, including

Interpolants in three or more dimensions: New generalizations of the above interpolants are presented. These have applications in scientific computing such as interpolation to measurements collected from 3D data sites. These data sites will be tessellated into tetrahedra and a piecewise polynomial surface will be defined over each of them, again with certain imposed continuity conditions. The Bernstein-Bézier form facilitates the construction and understanding of these interpolants considerably. Higher dimensional problems can be handled by an inductive generalization of the two- and three-dimensional interpolants.

Reflection lines for Bernstein-Bézier triangles: Reflection line simulation is a particularly effective surface interrogation tool. Its origins are in the car industry, where the aesthetics of a car body are judged by the reflection pattern generated by a set of parallel fluorescent light bulbs. Bernstein-Bézier patches lend themselves easily to the computation of reflection lines.

The solvability of interpolation problems: Interpolants over arbitrary triangulations are of interest in the field of scattered data interpolation. The dimensions of the underlying linear spaces are unknown. So of course B-spline-like basis functions are not known either. Some recent progress is described.[2]

References

1. G. Strang and F. Fix (1973): An Analysis of the Finite Element Method. Prentice Hall.

2. W. Böhm, G. Farin, J. Kahmann (1984): A Survey of Curve and Surface Methods in CAGD. Computer Aided Geometric Design, vol. 1, pp. 1-60.

N85-28578

A UNIFORM SUBDIVISION METHOD FOR TRIANGULAR BEZIER PATCHES

Kenneth I. Joy
Computer Science Division
Department of Electrical and Computer Engineering
University of California, Davis
Davis, California

1. Introduction

Bézier curves and tensor-product Bézier surface patches have been used extensively in the computer graphics/computer aided design community for the modeling of physical objects for automated design. In 1980, Lane and Riesenfeld [Lane-Riesenfeld, 1980] presented algorithms for the evaluation of Bézier curves and surfaces, and showed how these algorithms could be used in solving many of the geometric problems in CAD, and how they could be used in the computer graphic rendering problems for such surfaces. These algorithms were based upon an elegant subdivision technique, which was developed directly from the mathematical properties of the Bézier curve. This technique allows the manipulation of a curve/surface without the need to evaluate the blending functions and their derivatives.

Triangular Bézier patches have also been studied for the same reasons. Farin [Farin, 1982] has used them successfully in the interpolation of scattered 3-d data and Petersen [Petersen, 1984] has recently exhibited their use in a contouring algorithm. Subdivision of these patches has centered on a method [Farin, 1982] which subdivides the patch into three subpatches using intermediate control points calculated by the de Casteljau algorithm [de Casteljau, 1959]. However, due to the fact that one side of each of the three subtriangles is one of the sides of the parent triangle, the sides of the resulting subtriangles do not reduce uniformly in length. This implies that the polygonal rendering algorithms [Lane-Carpenter, 1979; Blinn, Carpenter, Lane, Whitted, 1980] (which require surfaces to be subdivided until they are flat) cannot be used with this subdivision method. (It is noted that Petersen used this method in his contouring algorithm, however, an extra subdivision step was added to insure that the lengths of the sides reduced uniformly.)

This paper presents a subdivision method for Bézier triangles that parallels the development of Lane-Riesenfeld for Bézier Curves. The method is developed from the mathematical description of the patch, and the resulting algorithm does not require evaluation of the blending functions. This method subdivides the triangle uniformly so that the lengths of the sides of the subtriangles are uniformly reduced. This method is a member of the class of methods first presented by Goldman [Goldman, 1983], which contain subdivision methods that arise from degenerate subdivision of Bézier tetrahedra. In this paper, we show that a certain method is a natural extension of the Lane-Riesenfeld results for Bézier curves.

2. Bézier Triangles

The Bézier triangle of degree n, defined by the set of control points $\bar{P} = \{P_{i,j,k} : i+j+k=n\}$ over the triangle T is given by

$$B_n(\bar{P}\,;T) = \sum_{i+j+k=n} P_{i,j,k}\, B_{i,j,k}(\mu_1,\mu_2,\mu_3)$$

where

$$B_{i,j,k}(\mu_1,\mu_2,\mu_3) = \frac{n!}{i!\,j!\,k!}\,\mu_1^i\,\mu_2^j\,\mu_3^k$$

are the bivariate Bernstein polynomials of degree n, using the barycentric coordinate system defined on the triangle T.

3. Subdivision of Bézier Triangles

Consider the triangle T with vertices P_1, P_2, P_3, where $P_i \in \mathbf{R}^2$. Let T be parameterized by the barycentric coordinates (μ_1,μ_2,μ_3). Consider the subdivision of T given by the following sets

$$T_1 = \left\{ P \in T : P = \mu_1 P_1 + \mu_2 P_2 + \mu_3 P_3, \text{ where } \mu_1 \geq \frac{1}{2},\ \mu_2 \geq 0,\ \mu_3 \geq 0 \right\}$$

$$T_2 = \left\{ P \in T : P = \mu_1 P_1 + \mu_2 P_2 + \mu_3 P_3, \text{ where } \mu_2 \geq \frac{1}{2},\ \mu_1 \geq 0,\ \mu_3 \geq 0 \right\}$$

$$T_3 = \left\{ P \in T : P = \mu_1 P_1 + \mu_2 P_2 + \mu_3 P_3, \text{ where } \mu_3 \geq \frac{1}{2},\ \mu_1 \geq 0,\ \mu_2 \geq 0 \right\}$$

$$T_4 = \left\{ P \in T : P = \mu_1 P_1 + \mu_2 P_2 + \mu_3 P_3, \text{ where } \mu_1 \leq \frac{1}{2},\ \mu_2 \leq \frac{1}{2},\ \mu_3 \leq \frac{1}{2} \right\}$$

The subdivision algorithm then proceeds as follows.

Subdivision Algorithm

Let $\bar{P} = \{P_{i,j,k} : i+j+k=n\}$ be a polygonal mesh defined over the triangle T and let

$$P_{i,j,k}^{[n_1 n_2 n_3]} = \begin{cases} \frac{1}{2}\left(P_{i,j,k}^{[n_1-1,n_2,n_3]} + P_{i-1,j+1,k}^{[n_1-1,n_2,n_3]}\right) & \text{if } n_1 > 0 \\ \frac{1}{2}\left(P_{i,j,k}^{[n_1,n_2,n_3-1]} + P_{i,j-1,k+1}^{[n_1,n_2,n_3-1]}\right) & \text{if } n_2 > 0 \\ \frac{1}{2}\left(P_{i,j,k}^{[n_1,n_2,n_3-1]} + P_{i+1,j,k-1}^{[n_1,n_2,n_3-1]}\right) & \text{if } n_3 > 0 \\ P_{i,j,k} & \text{if } n_1 = n_2 = n_3 = 0 \end{cases}$$

Then

ORIGINAL PAGE IS
OF POOR QUALITY

$$B_n(\overline{P}\,;T) = B_n(\overline{Q}_i\,;T_i),\ \text{for } i = 1, 2, 3, 4$$

where

$$\overline{Q}_1 = \left\{ P_{i,0,k}^{[n,0,k]} : i+k=n,\ m = 0, \ldots, i \right\}$$

$$\overline{Q}_2 = \left\{ P_{i,j,0}^{[i,j,m,0]} : i+j=n,\ m = 0, \ldots, j \right\}$$

$$\overline{Q}_3 = \left\{ P_{0,j,k}^{[0,j,k,m]} : j+k=n,\ m = 0, \ldots, k \right\}$$

$$\overline{Q}_4 = \left\{ P_{i,j,k}^{[i,j,k]} : i+j+k=n \right\}$$

Lemma

Let $\overline{Q} = \left\{ Q_{i,j,k} : i+j+k = n \right\}$ be the control points for the subdivided Bézier triangle over T_4; then we have

$$\begin{aligned} |\, Q_{i+1,j,k} - Q_{i,j,k} \,| &\leq M \\ |\, Q_{i,j+1,k} - Q_{i,j,k} \,| &\leq M \\ |\, Q_{i,j,k+1} - Q_{i,j,k} \,| &\leq M \end{aligned}$$

where

$$M = \frac{1}{2}\max \left\{ \begin{array}{l} |\, P_{i+1,j,k} - P_{i,j,k} \,| \\ |\, P_{i,j+1,k} - P_{i,j,k} \,| \\ |\, P_{i,j,k+1} - P_{i,j,k} \,| \end{array} : i+j+k = n, \text{and} i, j, k < n \right\}$$

4. Bibliography

[Bézier, 1972]. Bézier, P.E., ***Numerical Control -- Mathematics and Applications***, John Wiley and Sons, London, 1972.

[Blinn, Carpenter, Lane, Whitted, 1980]. Blinn, J.F., L.C. Carpenter, J.M. Lane, and T. Whitted, Scan Line Methods for Displaying Parametrically Defined Surfaces, CACM, Vol. 23, No. 1, 23-34.

[Boehm, 1984]. Boehm, W., G. Farin and J. Kahmann, "A survey of curve and surface methods in CAGD," Computer Aided Geometric Design 1(1), (1984), 1-60.

[de Casteljau, 1959]. de Casteljau, P. Courbes et Surfaces a Poles, S.A. Andre Citroen, Paris.

[Farin, 1980]. Farin, G., "Bézier Polynomials over Triangles," Technical Report 91, Department of Mathematics, Brunel University, Uxbridge, UK.

[Farin, 1982]. Farin, G. "Smooth Interpolation to Scattered 3D Data," ***Surfaces in Computer Aided Geometric Design***, R.E. Barnhill and W. Boehm (eds.), North-Holland Publishing Company, 1983, 43-63.

[Goldman, 1983]. Goldman, R.N., Subdivision Algorithms for Bézier Triangles, ***Computer Aided Design***, 15, 159-166.

[Joy, 1985]. Joy, K.I., Bézier Triangles, Technical Report CSE-85-3, Computer Science Division, Department of Electrical and Computer Engineering, University of California, Davis, California, January, 1985.

[Lane-Carpenter, 1979]. Lane, J.M. and L. Carpenter, A Generalized Scan Line Algorithm for the Computer Display of Parametrically Defined Surfaces, ***Computer Graphics and Image Processing***, 11, 1979, 290-297.

[Lane-Riesenfeld, 1980]. Lane, J.M. and R.F.Riesenfeld, A Theoretical Development for the Computer Generation and Display of Piecewise Polynomial Surfaces, ***IEEE Transactions on Pattern Analysis and Machine Intelligence***, Vol. PAMI-2, No. 1, January, 1980, 35-46.

[Petersen, 1984]. Petersen, C.S., Adaptive Contouring of Three-dimensional Surfaces, ***Computer Aided Geometric Design***, Vol. 1, July 1984, 61-74.

N85-28579

An Intuitive Approach to Geometric Continuity for Parametric Curves and Surfaces

(Extended Abstract)†

Tony D. DeRose
Brian A. Barsky

Berkeley Computer Graphics Laboratory
Computer Science Division
Department of Electrical Engineering and Computer Sciences
University of California
Berkeley, California 94720
U.S.A.

ABSTRACT

Parametric spline curves and surfaces are typically constructed so that some number of derivatives match where the curve segments or surface patches abut. If derivatives of up to order n are continuous, the segments or patches are said to meet with C^n, or n^{th} order *parametric continuity*. It has been shown previously that parametric continuity is sufficient, but not necessary, for geometric smoothness.

The geometric measures of *unit tangent* and *curvature* vectors for curves, and *tangent plane* and *Dupin indicatrix* for surfaces, have been used to define first and second order *geometric continuity*. In this work, we extend the notion of geometric continuity to arbitrary order n (G^n) for curves and surfaces, and present an intuitive development of constraints equations that are necessary and sufficient for it. The constraints result from a direct application of the univariate chain rule for curves, and the bivariate chain rule for surfaces. The constraints provide for the introduction of quantities known as *shape parameters*.

The approach we take is important for several reasons: First, it generalizes geometric continuity to arbitrary order for both curves and surfaces. Second, it shows the fundamental connection between geometric continuity of curves and geometric continuity of surfaces. Third, due to the chain rule derivation, constraints of any order can be determined more easily than derivations based exclusively on geometric measures.

1. Introduction

In recent years, computer-aided geometric design (CAGD) has relied heavily on mathematical descriptions of objects based on *parametric splines*. Spline curves are typically constructed by stitching together univariate parametric functions, requiring that some number of derivatives match at each *joint* (the points where the curve segments meet). If n derivatives agree at a given joint, the parametrizations there are said

This work was supported in part by the Defense Advanced Research Projects Agency under contract number N00039-82-C-0235, the National Science Foundation under grant number ECS-8204381, the State of California under a Microelectronics Innovation and Computer Research Opportunities grant, and a Shell Doctoral Fellowship.

†The complete paper appears in[5].

to meet with n^{th} *order parametric continuity* (C^n continuity for short). It has been previously demonstrated (see [2,7,8,11] for instance) that parametric continuity can be overly restrictive for many applications. To remedy this situation, another notion of continuity must be developed, one based on the geometry of the resulting curve or surface. We shall refer to this as *geometric continuity*.

It has recently come to our attention that many authors have independently defined this kind of continuity of first and second order (which we denote by G^1 and G^2, respectively) for curves and/or surfaces using geometric means. For curves, Fowler & Wilson[9], Sabin[13], Manning[11], Faux & Pratt[6], and Barsky[1] each independently defined first order continuity by requiring that the *unit tangent vectors* agree at the joints. To achieve second order continuity, both the unit tangent and *curvature* vectors were required to match. Nielson's ν-spline[12] possesses a similar kind of continuity. For surfaces, it is common to require matching of *tangent planes* for first order continuity (cf. Sabin[14] and Veron et al[15]). For surfaces of second order continuity, Veron et al and Kahmann[10] require continuity of *normal curvature* in every direction, at every point on the boundary shared by the constituent surface patches. As Veron et al and Kahmann each show, this is equivalent to requiring that the *Dupin indicatrix* of each patch agree at the boundary curve.

Although the geometric approaches described above are convenient and intuitive for first and second order continuity, a more algebraic development is better suited to the extension to continuity of higher order. The approach we take is based on the following simple idea:

> *P1: Don't base continuity on the parametrizations at hand; reparametrize, if necessary, to obtain parametrizations that meet with parametric continuity. If this can be done, the original parametrizations must also meet smoothly, at least in a geometric sense.*

The above concept is not a new one; similar principles have been discussed by Farin[7] and Veron et al[15]. What is new is the use of the principle to construct constraint equations (to be known as the *Beta constraints*) that are necessary and sufficient for geometric continuity of arbitrary order for both curves and surfaces. ‡

In this paper, we extend the notion of geometric continuity to arbitrary order n (G^n) and show (in a nonrigorous way) that the derivation of the Beta constraints results from a straightforward use of the univariate chain rule for curves and the bivariate chain rule for surfaces. This approach therefore provides new insight into the nature of geometric continuity and shows that geometric continuity of curves and surfaces need not be treated separately; the same basic principle of reparametrization applies to both. We also argue that, for first and second order continuity, the Beta constraints are equivalent to the geometric measures described above. However, due to chain rule derivation, the constraints are obtained with less effort using our method. For a more complete treatment, the reader is referred to Barsky & DeRose[3,5]

2. Geometric Continuity for Curves

We begin the study of geometric continuity for curves by examining the reparametrization process. Two parametrizations are said to be *GO-equivalent* if they have the same *geometry* (shape) and *orientation* (direction of tangent vector) at each point. Given a particular parametrization, all GO-equivalent parametrizations may be obtained by *functional composition*. More specifically, if $\mathbf{q}(u)$ and $\tilde{\mathbf{q}}(\tilde{u})$ are GO-equivalent, then they are related by $\tilde{\mathbf{q}}(\tilde{u}) = \mathbf{q}(u(\tilde{u}))$, for some appropriately chosen *change of parameter* $u(\tilde{u})$. Since $\mathbf{q}$ and $\tilde{\mathbf{q}}$ must have the same orientation, u must be an increasing function of $\tilde{u}$, implying that u must satisfy the *orientation preserving condition* $u^{(1)} > 0$ (in general, superscript (i) denotes the i^{th} derivative). A univariate parametrization is *regular* if the first derivative vector does not vanish. It is well known from differential geometry[6] that regularity is, in general, essential for the smoothness of the resulting curve. We will therefore restrict the discussion to regular parametrizations. We now give a more precise definition of G^n continuity:

‡ T. Goodman and L. Ramshaw have independently derived the univariate Beta constraints from the univariate chain rule.

Definition 1: *Let $\mathbf{r}(t), t \in [t_0, t_1]$ and $\mathbf{q}(u), u \in [u_0, u_1]$ be two parametrizations such that $\mathbf{r}(t_1) = \mathbf{q}(u_0)$ (see Figure 1). These parametrizations meet with G^n continuity at J if and only if there exist GO-equivalent parametrizations $\tilde{\mathbf{r}}(\tilde{t})$ and $\tilde{\mathbf{q}}(\tilde{u})$ that meet with C^n continuity.*

Definition 1 is simply a restatement of principle P1, but in practice one cannot examine all GO-equivalent parametrizations in an effort to find two that meet with parametric continuity. However, it is possible to find conditions on $\mathbf{r}$ and $\mathbf{q}$ that are necessary and sufficient for the *existence* of GO-equivalent parametrizations that meet with parametric continuity.

Due to the compositional structure of equivalent parametrizations, the derivation of the Beta constraints essentially reduces to an application of the chain rule. In particular, the chain rule is used to express derivatives of $\tilde{\mathbf{q}}$ in terms of derivatives of $\mathbf{q}$. For example, the Beta constraints for G^4 continuity for the situation shown in Figure 1 are:

$$
\begin{aligned}
\mathbf{r}^{(1)}(t_1) &= \beta_1 \mathbf{q}^{(1)}(u_0) \\
\mathbf{r}^{(2)}(t_1) &= \beta_1^2 \mathbf{q}^{(2)}(u_0) + \beta_2 \mathbf{q}^{(1)}(u_0) \\
\mathbf{r}^{(3)}(t_1) &= \beta_1^3 \mathbf{q}^{(3)}(u_0) + 3\beta_1\beta_2 \mathbf{q}^{(2)}(u_0) + \beta_3 \mathbf{q}^{(1)}(u_0) \\
\mathbf{r}^{(4)}(t_1) &= \beta_1^4 \mathbf{q}^{(4)}(u_0) + 6\beta_1^2\beta_2 \mathbf{q}^{(3)}(u_0) + (4\beta_1\beta_3 + 3\beta_2^2)\, \mathbf{q}^{(3)}(u_0) + \beta_4 \mathbf{q}^{(1)}(u_0).
\end{aligned}
\tag{2.1}
$$

where $\beta_1 > 0, \beta_2, \beta_3, \beta_4$ are arbitrary real numbers.

The right side of the first equation of (2.1) represents the derivative of $\tilde{\mathbf{q}}$ written in terms of the derivative of $\mathbf{q}$, where the substitution $u^{(1)} = \beta_1$ has been made. Thus, β_1 determines, to first order, the function $u(\tilde{u})$ in the neighborhood of the joint. Similarly, the right side of the second equation of (2.1) represents the second derivative of $\tilde{\mathbf{q}}$ expressed in terms of derivatives of $\mathbf{q}$, where $u^{(2)}$ has been given the value β_2 at the joint. This process is continued for the higher order constraints.

Although the first two equations of (2.1) were derived using the chain rule, they are identical to the constraints resulting from a geometric derivation of unit tangent and curvature vector continuity, respectively[3,11]. Thus, our approach reduces to previous definitions of G^1 and G^2 continuity.

In general, a property of the chain rule can be used to easily show that for n^{th} order geometric continuity for curves, n *shape parameters* $\beta_1, \ldots, \beta_n$ are introduced[3]. The quantities $\beta_1, \ldots, \beta_n$ are called shape parameters because they can be made available to a designer in a CAGD environment to change the shape of curves and surfaces. Since geometric continuity provides for the introduction of shape parameters, it may be desirable to generalize existing spline techniques to obtain their geometrically continuous analogues. For instance, the Beta-spline[1,2] is an *approximating* technique that possesses shape parameters; an *interpolating* technique is described in DeRose & Barsky[4]. Faux & Pratt[6] and Farin[7] use the extra freedom allowed by geometric continuity to place *Bézier control vertices*.

3. Geometric Continuity for Surfaces

A parametric surface patch is defined by a bivariate function such as $\mathbf{G}(u, v) = (x(u, v), y(u, v), z(u, v))$, where u and v are allowed to range over some region of the uv plane. Such a parametrization is *regular* if the first order partial derivatives are linearly independent. N^{th} order parametric continuity of two surface patches requires that all like partial derivatives of order up to n agree for each point of the boundary curve. Superscript (i, j) will be used to denote the i^{th} partial with respect to the first variable, and the j^{th} partial with respect to the second. Just as for curves, parametric continuity is sufficient for geometric smoothness, but can be overly restrictive.

The notion of reparametrization as a basis for the determination of continuity can readily be extended to surfaces by making a definition analogous to Definition 1. The *bivariate chain rule* can then be used to determine constraint equations. However, instead of shape parameters being introduced, a property of the bivariate chain rule shows that $n(n+3)$ *shape functions* are introduced when two surface patches are stitched

together with G^n continuity. Referring to the situation of Figure 2 and using the bivariate chain rule, one can show that $\mathbf{F}(u,v)$ and $\mathbf{G}(s,t)$ meet with G^1 continuity if and only if

$$\begin{aligned} \mathbf{F}^{(0,1)} &= \beta_{00}\,\mathbf{G}^{(0,1)} + \beta_{01}\,\mathbf{G}^{(1,0)} \\ \mathbf{F}^{(1,0)} &= \beta_{10}\,\mathbf{G}^{(0,1)} + \beta_{11}\,\mathbf{G}^{(1,0)} \end{aligned} \tag{3.1}$$

holds for each point $\mathbf{P}$ of γ where the ***shape functions*** $\beta_{00}, \beta_{01}, \beta_{10}, \beta_{11}$ satisfy the orientation preserving condition $\beta_{00}\beta_{11} - \beta_{10}\beta_{01} > 0$. The shape functions determine how derivatives of $\mathbf{G}$ are related to derivatives of a GO-equivalent parametrization ($\tilde{\mathbf{G}}$) that meets $\mathbf{F}$ with first order parametric continuity. Although equations (3.1) were derived from the bivariate chain rule, they also have geometric significance. More specifically, they are necessary and sufficient conditions for ***tangent plane continuity*** between $\mathbf{F}$ and $\mathbf{G}$. Thus, the abstract algebraic approach of reparametrization and the chain rule agrees with geometric intuition for first order continuity of surfaces.

It can be shown that the constraints resulting from the chain rule approach are equivalent to requiring that the Dupin indicatrix of the patches match along the boundary curve. Thus, the chain rule approach agrees with geometric intuition for both G^1 and G^2 continuity. Moreover, the chain rule approach yields the second order constraints with less effort than the geometric approach. For higher order continuity, geometric intuition becomes more feeble, but the chain rule approach still applies.

4. Conclusion

We have defined n^{th} order geometric continuity for parametric curves and surfaces, and derived the Beta constraints that are necessary and sufficient for it. The derivation of the Beta constraints is based on a simple principle of reparametrization in conjunction with the univariate chain rule for curves, and the bivariate chain rule for surfaces. This approach therefore uncovers the connection between geometric continuity for curves and geometric continuity for surfaces, provides new insight into the nature of geometric continuity in general, and allows the determination of the Beta constraints with less effort than previously required.

The use of the Beta constraints allows the introduction of n shape parameters for curves, and $n(n+3)$ shape functions for surfaces. The shape parameters and shape functions may be used to modify the shape of a geometrically continuous curve or surface, respectively. However, geometric continuity is only appropriate for applications where the particular parametrization used is unimportant since parametric discontinuities are allowed.

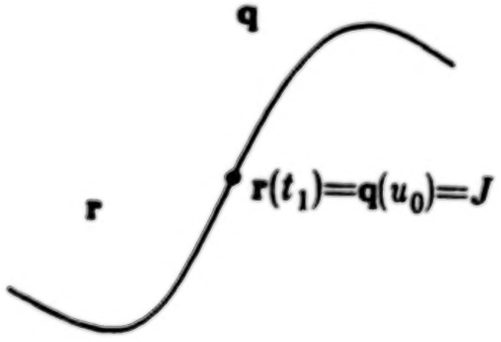


Figure 1: *The parametrizations* r *and* q *meet at the common point* $\mathbf{r}(t_1) = \mathbf{q}(u_0)$.

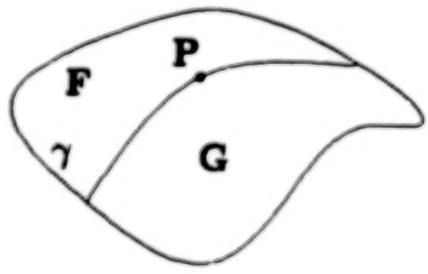


Figure 2: *The surface patches created by* F *and* G *meet at the boundary curve* γ.

References

1. Brian A. Barsky, *The Beta-spline: A Local Representation Based on Shape Parameters and Fundamental Geometric Measures*, Ph.D. Thesis, University of Utah, Salt Lake City, Utah (December, 1981).

2. Brian A. Barsky, *Computer Graphics and Geometric Modelling Using Beta-splines*, Springer-Verlag, Tokyo, 1985.

3. Brian A. Barsky and Tony D. DeRose, *Geometric Continuity for Parametric Curves*, Technical Report No. UCB/CSD 84/205, University of California, Berkeley (October, 1984).

4. Tony D. DeRose and Brian A. Barsky, "Geometric Continuity and Shape Parameters for Catmull-Rom Splines (Extended Abstract)," pp. 57-62 in *Proceedings of Graphics Interface '84*, Ottawa (27 May - 1 June, 1984).

5. Tony D. DeRose and Brian A. Barsky, "An Intuitive Approach to Geometric Continuity for Parametric Curves and Surfaces," pp. 343-351 in *Proceedings of Graphics Interface '85*, Montreal (27-31 May, 1985).

6. Manfred P. DoCarmo, *Differential Geometry of Curves and Surfaces*, Prentice-Hall, Inc., Englewood Cliffs, New Jersey (1976).

7. Gerald Farin, "Visually C^2 Cubic Splines," *Computer-Aided Design*, Vol. 14, No. 3, May, 1982, pp. 137-139.

8. Ivor D. Faux and Michael J. Pratt, *Computational Geometry for Design and Manufacture*, Ellis Horwood Ltd. (1979).

9. A. H. Fowler and C. W. Wilson, "Cubic Spline, A Curve Fitting Routine," Union Carbide Corporation Report, Y-1400 (Rev. I.) (1966).

10. Juergen Kahmann, "Continuity of Curvature Between Adjacent Bézier Patches," pp. 65-75 in *Surfaces in Computer Aided Geometric Design*, ed. Robert E. Barnhill and Wolfgang Boehm, North-Holland Publishing Company (1983).

11. J. R. Manning, "Continuity Conditions for Spline Curves," *The Computer Journal*, Vol. 17, No. 2, May, 1974, pp. 181-186.

12. Gregory M. Nielson, "Some Piecewise Polynomial Alternatives to Splines under Tension," pp. 209-235, in *Computer Aided Geometric Design*, ed. Robert E. Barnhill and Richard F. Riesenfeld, Academic Press, New York (1974).

13. Malcolm A. Sabin, *Parametric Splines in Tension*, Technical Report No. VTO/MS/160, British Aircraft Corporation, Weybridge, Surrey, England (July 23, 1970).

14. Malcolm A. Sabin, *The Use of Piecewise Forms for the Numerical Representation of Shape*, Ph.D. Thesis, Budapest (1976).

15. M. Veron, G. Ris, and J.-P. Musse, "Continuity of Biparametric Surface Patches," *Computer-Aided Design*, Vol. 8, No. 4, October, 1976, pp. 267-273.

N85-28580

URN MODELS AND BETA-SPLINES

R. N. Goldman
Control Data Corporation
Minneapolis, Minnesota

1. Introduction

A well-established connection exists between discrete urn models and the standard curves and surfaces used in computer-aided geometric design (CAGD) [1],[2], [3],[4],[5]. The Bézier and B-spline blending functions both model elementary stochastic processes, and many of the geometric properties of Bézier and B-spline curves and surfaces can be derived by studying these probabilistic models [4],[5]. Recently Barsky has introduced a new type of spline into CAGD called the beta-spline [6],[7],[8]. The purpose of this paper is to try to gain some insight into the properties of beta-splines by applying the techniques of urn models.

2. Beta-Splines

Beta-splines are generalizations of B-splines. They were developed in order to replace the somewhat artificial concept of parametric continuity by the more natural notion of geometric continuity. Briefly the idea is this: Two curves L(t) $t_0 \le t \le t_1$, R(u) $u_0 \le u \le u_1$ are said to meet with n^{th}-order parametric continuity (C^n) if and only if

$$\left.\frac{d^k R}{du^k}\right|_{u=u_0} = \left.\frac{d^k L}{dt^k}\right|_{t=t_1} \qquad k = 0,1,\ldots,n$$

Unfortunately this definition depends on more than just the geometry of the curves L(t),R(u); it also depends on the specific choice of their parametric representations. A linear change of parameter $u=\beta v$ $\beta>0$ will not change the shape of the curve R(u), but by the chain rule

$$\left.\frac{d^k R}{dv^k}\right|_{v=v_0} = \beta^k \left.\frac{d^k R}{du^k}\right|_{u=u_0} = \beta^k \left.\frac{d^k L}{dt^k}\right|_{t=t_1} \neq \left.\frac{d^k L}{dt^k}\right|_{t=t_1}$$

Thus R(βv) and L(t) do not meet with n^{th}-order parametric continuity even though the curves R(βv),R(u) are geometrically identical. To rectify this anomaly, the concept of geometric continuity is introduced.

Two curves L(t),R(u) are said to meet with linear n^{th}-order geometric continuity (LG^n) if and only if there exists a constant $\beta>0$ such that

$$\left.\frac{d^k R}{du^k}\right|_{u=u_0} = \beta^k \left.\frac{d^k L}{dt^k}\right|_{t=t_1} \qquad k = 0,1,\ldots,n$$

PRECEDING PAGE BLANK NOT FILMED

It is easy to check that the notion of linear n^{th}-order geometric continuity is invariant under linear changes of parameter. Of course, this concept is not invariant under non-linear changes of parameter. A more general notion of geometric continuity (G^n) and more general constraint equations invariant under non-linear changes of parameter are given in [8].

Splines have typically been defined in terms of parametric continuity, and the B-splines form a convenient basis for these parametric splines. The more general notion of geometric continuity requires us to search for a new set of basis functions suitable for these new types of splines. These basis functions are called beta-splines. We shall now use urn models to construct beta-splines and study their properties.

3. An Urn Model for Beta-Splines

Consider an urn initially containing w white balls and b black balls. One ball at a time is drawn at random from the urn, its color inspected, and then returned to the urn. If the ball was the j^{th} white ball to be chosen, then $\beta^j(w+b)$ additional black balls are added to the urn; if the ball was the j^{th} black ball to be chosen, then $\beta^{-j}(w+b)$ additional white balls are added to the urn.

We now introduce the following notation:

$t = \frac{w}{w+b}$ = probability of selecting a white ball on the first trial

$\sigma_j(\beta) = 1 + \beta + \ldots + \beta^{j-1}$

$s_j^N(t) = s_j^N(\beta,t)$ = probability of selecting a white ball after selecting exactly j white balls in the first N trials

$f_j^N(t) = f_j^N(\beta,t)$ = probability of selecting a black ball after selecting exactly j white balls in the first N trials

$B_j^N(t) = B_j^N(\beta,t)$ = probability of selecting <u>exactly</u> j white balls in the first N trials

For each fixed β it can be shown that the functions $B_0^N(t),\ldots,B_N^N(t)$ are linearly independent polynomials of degree N and they satisfy the constraint equations(*)

$$(*)\quad \frac{d^k B_{j-1}^N}{dt^k}\Big|_{t=0} = \beta^k \frac{d^k B_j^N}{dt^k}\Big|_{t=1} \qquad k = 0,1,\ldots,N-1$$

The functions $B_0^N(t),\ldots,B_N^N(t)$ are the beta-spline basis functions. If $\beta=1$, these functions are the uniform B-spline basis functions and the urn model is the standard urn model for B-splines [1], [4].

Given a sequence of control points $P=(P_0,\ldots,P_M)$, we can use these beta-spline basis functions as blending functions to construct LC^{N-1} continuous beta-spline curves in much the same way that we use the uniform B-spline basis functions to define C^{N-1} continuous B-spline curves. Define the i^{th} curve

segment by setting

$$B_i[\beta,P](t) = \sum_{j=0}^{N} B_j^N(\beta,t-i)P_{i+j} \qquad i \le t \le i+1$$

and define the beta-spline curve by setting

$$B[\beta,P](t) = B_i[\beta,P](t) \qquad i \le t \le i+1 \qquad 0 \le i \le M-N$$

From the constraint equations (*) it follows immediately that

$$\left.\frac{d^k B_{i+1}[\beta,P]}{dt^k}\right|_{t=i+1} = \beta^k \left.\frac{d^k B_i[\beta,P]}{dt^k}\right|_{t=i+1} \qquad k = 0,1,\ldots,N-1$$

Thus $B[\beta,P](t)$ is an LG^{N-1} continuous beta-spline curve.

Without moving the control points, we can alter the shape of the beta-spline curve $B[\beta,P](t)$ simply by changing the scalar parameter β. The effect of increasing β is to move the curve closer to its control polygon and to bias the curve towards its initial control points. Thus our β corresponds to Barsky's bias parameter β_1 [7].

In table 1 we summarize those properties of beta-spline basis functions and beta-spline curves which are directly derivable from the beta-spline urn model. Many of these properties are new and are presented here for the first time.

4. Conclusion

Urn models can be used to construct beta-spline basis functions and to derive the basic properties of these blending functions and the corresponding beta-spline curves. This is only the beginning; much work remains to be done. Here we have dealt only with the simple notion of linear geometric continuity and with the most elementary beta parameter. Non-linear geometric continuity leads to additional beta parameters and to more complicated basis functions [8]. Whether urn models can give us any insight into these higher order concepts still remains to be investigated.

References

1. Goldman, R. N., An Urnful of Blending Functions, IEEE Computer Graphics and Applications, vol. 3, no. 7, 1983, pp. 49-54.

2. Goldman, R. N., An Intuitive Approach to Bézier and Other Random Curves and Surfaces, SIGGRAPH Tutorial on Freeform Curves and Surfaces, 1983.

3. Goldman, R. N., Geometry and Probability, SIGGRAPH Tutorial on Freeform Curves and Surfaces, 1984.

4. Goldman, R. N., Polya's Urn Model and Computer Aided Geometric Design, SIAM Journ. on Algebraic and Discrete Methods, vol. 6, no. 1, 1985, pp. 1-28.

5. Goldman, R. N., Urn Models, Approximations, and Splines, Submitted to the Journal of Approximation Theory.

6. Barsky, B. A., The Beta-Spline: A Local Representation Based on Shape Parameters and Fundamental Geometric Measures, Ph.D. Thesis, University of Utah, Salt Lake City, Utah, 1981.

7. Barsky, B. A. and Beatty, J. C., Local Control of Bias and Tension in Beta-Splines, ACM Trans. on Graphics, vol. 2, no. 2, 1982, pp. 109-134.

8. Barsky, B. A. and DeRose, T. D., Geometric Continuity of Parametric Curves, Tech. Rept. No. UCB/CSD 84/205, Computer Science Division University of California, Berkeley, October 1984.

TABLE 1 - PROPERTIES OF BETA-SPLINE BASIS FUNCTIONS AND CURVES

	Urn	Basis Functions	Curve
1.	Probability Interpretation	$\Rightarrow \sum_{j=0}^{n} B_j^n(t) = 1$	$\Rightarrow$ Coordinate Free
		$\Rightarrow B_j^n(t) \geq 0 \quad 0 \leq t \leq 1$	$\Rightarrow$ Local Convex Hull Property
2.	Symmetry between white, β and black, β^{-1}	$\Rightarrow$ Symmetry Formula $B_j^n(\beta,t) = B_{n-j}^n(\beta^{-1},1-t)$	$\Rightarrow$ Curve Symmetry $R(\beta,P)(t) = R(\beta^{-1},P_{\text{reverse}})(1-t)$
3.	Counting	$\Rightarrow$ Explicit Formulas	$\Rightarrow$ Recursion Formula
			$\Rightarrow$ Locality of End Points
		$B_j^n(t) = \dfrac{\beta^{n-j}(\sigma_j(\beta)t)^j(\sigma_{n+1}(\beta)-t)^{n-j}}{\sigma_{n+1}(\beta)}$	
			$\Rightarrow$ Explicit Recursion Formula (case 4)
		$B_j^n(t) = \dfrac{\beta^{n-j}t^j + \sigma_{n-j}(\beta)}{\sigma_{n+1}(\beta)}$	
		$B_0^n(t) = \dfrac{\beta^{n(n-1)/2}(1-t)^n}{\prod_{j=1}^{n}\sigma_j(\beta)}$	$\Rightarrow$ P_j does not affect $\left.\dfrac{d^k R(\beta,P)}{dt^k}\right\|_{t=0}$ $k=0,1,\ldots,n-1$
		$B_n^n(t) = \dfrac{t^n}{\prod_{j=1}^{n}\sigma_j(\beta)}$	$\Rightarrow$ $P_{j\neq n}$ does not affect $\left.\dfrac{d^k R(\beta,P)}{dt^k}\right\|_{t=1}$ $k=0,1,\ldots,n-1$
4.	Relationship between first n and first $n+1$ picks	$\Rightarrow$ Recursion Formula $B_j^{n+1}(t) = \delta_j^n(t)B_j^n(t) + \omega_{j-1}^n(t)B_{j-1}^n(t)$	$\Rightarrow$ Geometric Construction Algorithm Let $0 \leq t \leq 1$ $P_j^0(t) = P_j \quad 0 \leq j \leq n$ $P_j^k(t) = \delta_j^{k-1}(t)P_{j+1}^{k-1}(t) + \omega_j^{k-1}(t)P_j^{k-1}(t)$ Then $R(\beta,P)(t) = P_0^n(t)$
5.	Recursion Formula	$\Rightarrow$ Polynomial Functions	$\Rightarrow$ Polynomial Spline
		$\Rightarrow$ Differentiability Constraints $\left.\dfrac{d^k B_j^n}{dt^k}\right\|_{t=0} = \beta^k \left.\dfrac{d^k B_{j-1}^n}{dt^k}\right\|_{t=1}$	$\Rightarrow$ C^{n-1}
6.	Linear Independence of first $n+1$ σ-moments	$\Rightarrow$ Linear Independence	$\Rightarrow$ Locally Non-Degenerate
		$\Rightarrow$ Polynomial Basis	$\Rightarrow$ Local Subdivision Algorithm
7.	Limiting Conditions	Limits	Tension/Bias
	If $\beta = \infty$ at most 1 white ball can be selected.	$\Rightarrow$ $\lim_{\beta\to\infty} B_j^n(t) = (1-t)^n \quad j=0$; $= 1-(1-t)^n \quad j=1$; $= 0 \quad j\neq 0,1$	$\Rightarrow$ $\lim_{\beta\to\infty} R(\beta,P)(t)$ is the polygon determined by $P_0,\ldots,P_{n-1}$.
	If $\beta = 0$ at most 1 black ball can be selected	$\Rightarrow$ $\lim_{\beta\to 0} B_j^n(t) = t^n \quad j=n$; $= 1-t^n \quad j=n-1$; $= 0 \quad j\neq n-1,n$	$\Rightarrow$ $\lim_{\beta\to 0} R(\beta,P)(t)$ is the polygon determined by $P_{n-1},\ldots,P_n$.
8.	Adding only balls of the opposite color	$\Rightarrow$ Total Positivity	$\Rightarrow$ Variation Diminishing Property
9.	Two urns each with two colors	$\Rightarrow$ Independent Distributions $B_{ij}^{nn}(s,t) = B_i^n(s)B_j^n(t)$	$\Rightarrow$ Rectangular Tensor Product Surfaces

ORIGINAL PAGE IS
OF POOR QUALITY

N85-28581

PARAMETERIZATION IN GRID GENERATION*

C. Wayne Mastin
Mississippi State University
Mississippi State, MS

The mathematical description of a physical object is an absolute necessity in solving nearly any problem in computational fluid dynamics or related fields where one must compute the numerical solution of partial differential equations. The selection of points on which to compute a numerical simulation differs from the geometry definition procedures in computer-aided design. Whereas in the latter case decisions are often based on aesthetics, the distribution of grid points for calculating the solution of partial differential equations must be chosen so as to include consideration of truncation error, stability, and the resolution of the solution near boundary layers and shocks (ref. 1). It is therefore important to be able to specify the distribution of points along a grid line.

The problem of distributing points along a curve will now be considered. It will be assumed that the curve is defined parametrically. The objective is to select a set of parameter values so that the corresponding points on the curves are properly distributed. The distribution may be based on some intrinsic property of the curve such as arc length or curvature.

Suppose a curve is given parametrically by the equation

$$r = r(\eta),\ 0 \leq \eta \leq 1$$

where $r = (x,y,z)$. The desired set of values for η will be defined by introducing a reparameterization of the curve

$$r = r(\eta(t)),\ 0 \leq t \leq 1$$

For each value of t, the arc length derivative $d(t)$ will be defined so that

$$[d(t)]^2 = r_t \cdot r_t$$

The function $d(t)$ cannot be completely arbitrary since it must satisfy

$$\int_0^1 d(t)dt = L \qquad (1)$$

where L is the length of the curve. Since r is a composite function of t, we also have

$$r_t \cdot r_t = (r_\eta \cdot r_\eta)\eta_t^2$$

---

*Research supported by NASA Langley Research Center under Grant No. NSG 1577.

PRECEDING PAGE BLANK NOT FILMED

Therefore, when $d(t)$ is given, the function $\eta(t)$ is the solution of the initial value problem

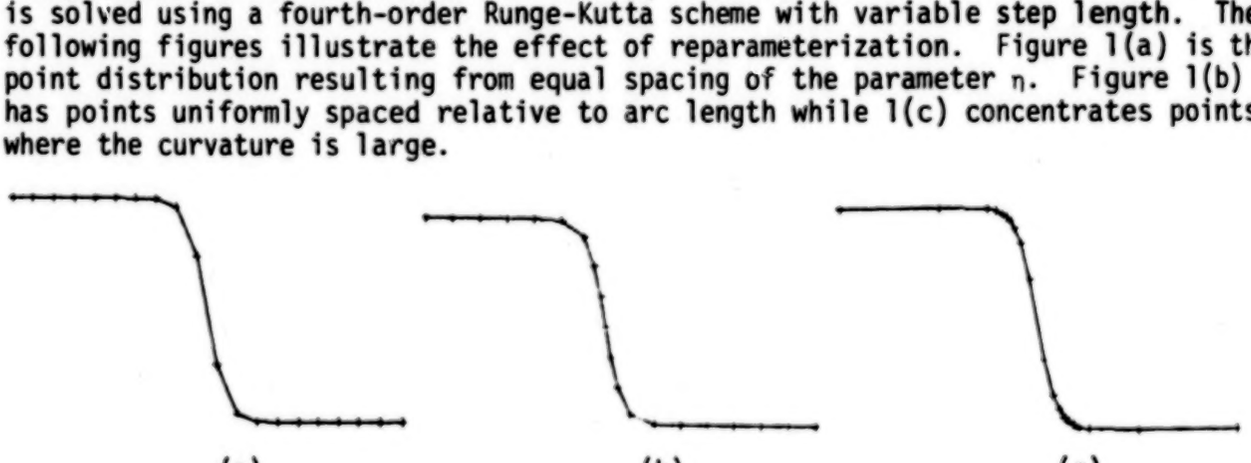
$$\eta_t = \frac{d(t)}{(r_\eta \cdot r_\eta)^{1/2}} = F(t,\eta),\ \eta(0) = 0 \qquad (2)$$

This problem can be solved accurately and efficiently by various numerical algorithms. It can be further noted that stability will be enhanced if $r_\eta \cdot r_\eta$ is an increasing function of η, that is, grid spacing increases with uniform increments of η. The numerical solution of this initial value problem may not exactly satisfy the condition $\eta(1) = 1$. This may be due to error in the numerical solution or the fact that $d(t)$ is not exactly normalized by the above integral condition (1). In either case, the solution is computed until the value $\eta = 1$ is reached, and then the independent variable t is scaled so that, as a function of the new variable, $\eta(1) = 1$. This scaling process will alter the grid spacing, but the ratio of the grid spacing at any two points will be unchanged. Thus, if only the relative spacings at points along the curve are to be prescribed, no normalization of $d(t)$ is necessary. An example would be the case when equal spacing of grid points along the curve is desired. Any constant value for $d(t)$ would suffice.

The parameterization algorithm has been used to distribute points along various plane curves. For the first example, consider the curve defined by

$$y = \tanh(5x)\ ,\ -2 \leq x \leq 2$$

The parameter η is introduced, with $x = 4\eta - 2$, and the initial value problem (2) is solved using a fourth-order Runge-Kutta scheme with variable step length. The following figures illustrate the effect of reparameterization. Figure 1(a) is the point distribution resulting from equal spacing of the parameter η. Figure 1(b) has points uniformly spaced relative to arc length while 1(c) concentrates points where the curvature is large.



(a) (b) (c)

Figure 1. Grid point distributions for (a) uniform η, (b) $d(t) = c$, c = constant, and (c) $d(t) = c/(1 + 5|\kappa|)$, κ = curvature.

The above example was selected because the graph is typical of solutions for problems with shocks. In such cases a grid as in 1(c) will minimize smearing or oscillations in the solution.

The second example applies to the construction of a two-dimensional grid between an ellipse and a circle. On each grid line connecting the boundary components, a function of the form

$$d(t) = ae^{t} + be^{-t}$$

is used with the constants a and b selected so that (1) holds and also the grid spacing at the elliptical $t = 0$ boundary component is some specified value. An example of such a grid, with a small uniform spacing at the ellipse, is indicated in Figure 2. In this example a curvature-based reparameterization was also used to redistribute grid points along the ellipse. Finally, Figure 3 contains the plot of the same region with a grid having uniform spacing along every grid line. This was accomplished in an iterative process with the reparameterization performed in alternating coordinate directions.

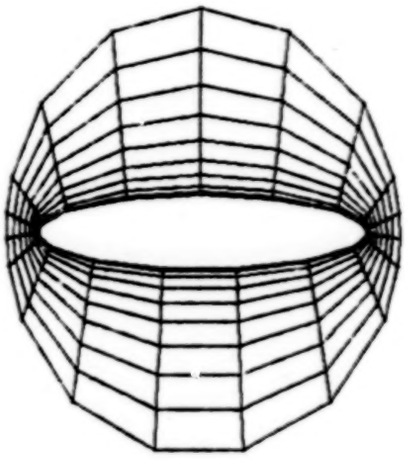


Figure 2. Grid concentration near inner boundary.

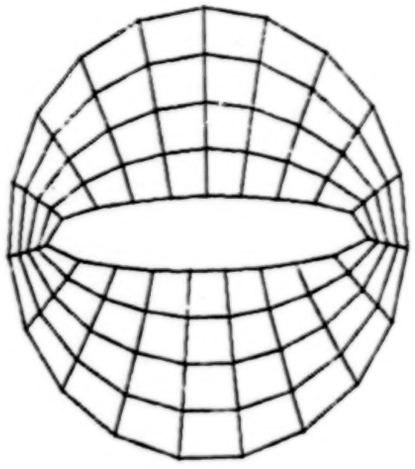


Figure 3. Uniform spacing in both directions.

Many of the potential applications of reparameterization in grid generation have not been addressed. One particularly attractive possibility would be to derive a method for generating adaptive grids by selecting d(t) to be a solution dependent function. Of course, with this or any grid redistribution scheme the skewness of the grid lines and the overall smoothness of the grid should be examined. A detailed study of the effect of grid properties on the numerical solution of partial differential equations may be found in the following reference.

REFERENCE

1. Thompson, J. F.; Warsi, Z.U.A.; and Mastin, C. W.: Numerical Grid Generation, Elsevier/North-Holland, New York, 1985.

N85-28582

CONTOURING TRIVARIATE DATA

Sarah E. Stead
NASA Ames Research Center
Moffett Field, CA

George T. Makatura
Informatics General Corporation
Palo Alto, CA

ABSTRACT

In many applications, the data consist of discrete 3-D points at which one or more parameters are given. To display contours, the data are represented by a continuous function which is evaluated at any point as needed for contouring. We present contouring results applicable both to artitrarily spaced data and to data which lie on a topologically rectangular three-dimensional grid. We assume that the contours are to be described by 3-D display lists for viewing on a dynamic color graphics device; that is, they will not simply be projected into 2-D and viewed as a static image on a frame buffer. Dynamic viewing of color contours may be essential to the proper interpretation of results.

We assume that the gridded data lie on a topologically rectangular grid although two or more nodes may be the same point. This type of gridded data is common in computational fluid dynamics. Parametric tensor product methods may be used to fit the gridded data and to generate the contours. Rectangular elements are convenient but not necessary. For example, there are other methods [1] which are effective for contouring over tetrahedral elements.

Various types of contours are helpful in interpreting trivariate data. Since we are interested in displays which can be viewed dynamically, we define contours as 3-D polygons, which are shaded by using polygon fill, or as 3-D vectors. Cross sections through the grid can be contoured using vectors or polygons (Figure 1). Contours over all grid layers are computed as vectors (Figure 2) or polygons (Figure 3). By combining adjoining grid cells into one cell, the amount of display list data needed to define the contours can be reduced. This can either speed up the dynamics or allow more contours (or other data) to be displayed. If a static picture is sufficient, the polygons can be scan-line rendered with transparency (Figure 3). An unusual feature that we developed consists of drawing the contours for one parameter onto the shape of a contour of a different parameter (Figure 4). In Figure 4, the shape is the 0.1 contour of a tricubic function and the gray bands indicate the derivative with respect to x of the tricubic function.

Multistage methods [2,3] for arbitrarily spaced trivariate data allow for efficient computation of contours. In these methods, an interpolant or approximant is fit to the arbitrarily spaced data and then evaluated at the nodes of a rectangular grid. The gridded data are then contoured as discussed above. One area where multistage methods have been applied is in geology, where the data consist of pollen counts at various depths in cores drilled in the earth. An analogous example occurs in well log data from an oil field in which various parameters are measured at many depths in each oil well. A similar application occurs in experimental fluid dynamics where pressure is recorded at various points on an

PRECEDING PAGE BLANK NOT FILMED

aircraft body during wind tunnel tests. Since these data occur along a surface, rather than throughout a volume, different representation methods are required, but the same display techniques are relevant.

REFERENCES

1. Peterson, C. S., Contours of Three and Four Dimensional Surfaces. Masters Thesis, Dept. of Math., Univ. of Utah, 1983.

2. Barnhill, R. E., and Stead, S. E., Multistage Trivariate Surfaces. Rocky Mountain J. of Math., vol. 14, no. 1, winter 1984, pp. 103-118.

3. Stead, S. E., Smooth Multistage Multivariate Approximation. Ph.D. Thesis, Div. of Appl. Math., Brown Univ., Providence, RI, 1983.

ORIGINAL PAGE IS
OF POOR QUALITY

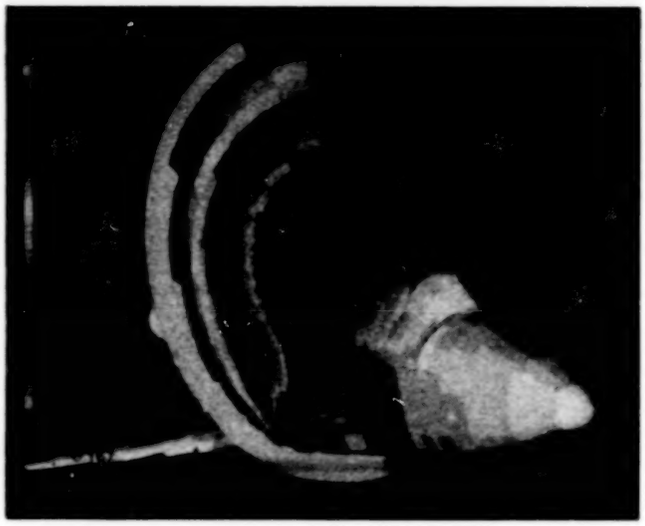


Figure 1. Density contours on two cross sections.

Figure 2. Pressure contours.

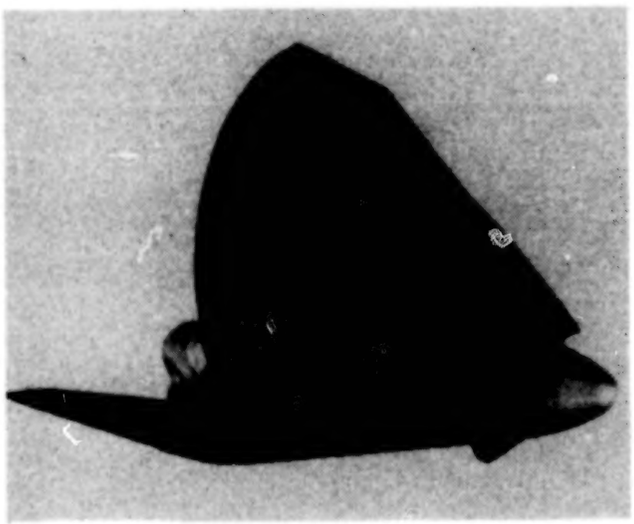


Figure 3. Pressure contours using transparent polygons.

Figure 4. Contouring on a contour.

N85-28583

GENERATION OF SURFACE COORDINATES BY ELLIPTIC PARTIAL DIFFERENTIAL EQUATIONS*

Z.U.A. Warsi

Department of Aerospace Engineering
Mississippi State University
Mississippi State, MS 39762

The problem of generating spatial coordinates by numerical methods through carefully selected mathematical models is of current interest both in mechanics and physics. A review of various methods of coordinate generation in two and three-dimensional Euclidean spaces is available in reference 1, and reference may also be made to the proceedings of two recent conferences (references 2 and 3) and a book (ref. 4) on the topic of numerical grid generation.

In this paper the problem of generation of a desired system of coordinates in a given surface has been considered which essentially is an effort directed to the problem of grid generation in a two-dimensional non-Euclidean space. The mathematical model selected for this purpose is based on the formulae of Gauss for a surface and has been discussed by the author in earlier publications (refs. 5 - 9).

The formulae of Gauss for a surface ν=const. can be written compactly by using the summation convention on repeated lower and upper indices as

$$\underset{\sim}{r}_{,\alpha\beta} = T^{\delta}_{\alpha\beta}\,\underset{\sim}{r}_{,\delta} + \underset{\sim}{n}^{(\nu)} b_{\alpha\beta} \tag{1}$$

where $\underset{\sim}{r}=(x,y,z)$, $T^{\delta}_{\alpha\beta}$ are the surface Christoffel symbols, $\underset{\sim}{n}^{(\nu)}$ is the unit surface normal on the surface ν=const., $b_{\alpha\beta}$ are the coefficients of the second fundamental form, and a comma denotes the partial derivative with respect to the surface coordinates x^{α}(ν and the other Greek indices assuming cyclic values). Inner multiplication by $G_{\nu}g^{\alpha\beta}$ of equation (1) yields

$$D\underset{\sim}{r} + G_{\nu}(\Delta_2^{(\nu)} x^{\delta})\,\underset{\sim}{r}_{,\delta} = \underset{\sim}{n}^{(\nu)} R, \tag{2}$$

where

$$G_{\nu} = g_{\alpha\alpha}g_{\beta\beta} - (g_{\alpha\beta})^2, \quad \nu,\alpha,\beta \text{ cyclic},$$

$$D = G_{\nu}g^{\alpha\beta}\partial_{\alpha\beta}, \tag{3a-c}$$

$$R = G_{\nu}g^{\alpha\beta}b_{\alpha\beta} = G_{\nu}\,(k_I^{(\nu)} + k_{II}^{(\nu)}),$$

*Research performed under grant AFOSR-0185.

PRECEDING PAGE BLANK NOT FILMED

and

$$\Delta_2^{(\nu)} x^{\alpha} = -g^{\beta\gamma} T^{\alpha}_{\beta\gamma}, \tag{3d}$$

is the second-order differential parameter of Beltrami. In equation (3c), $k_I^{(\nu)}$ and $k_{II}^{(\nu)}$ are the principal curvatures at a point on the surface ν=const.

Equation (2) provides three coupled quasilinear elliptic partial differential equations with the Cartesian coordiantes x,y,z as dependent variables. These equations are nonhomogenous with the right hand sides depending on the components of both the normal and the mean curvature of the surface, thus reflecting some geometrical aspects of the surface in an explicit manner.

Some of the problems listed below have been solved successfully by taking equation (2) as a basic differential model.

I. If the mathematical equation of the surface is available in the form $F(x,y,z)=0$, then equation (2) can be used to introduce any desired coordinate system in the surface. (For discrete x,y,z values of a surface, the form $F(x,y,z)=0$ can be obtained either by a least square or piecewise approximation method. Knowing $F(x,y,z) = 0$, one can find $k_I^{(\nu)} + k_{II}^{(\nu)}$ as a function of x,y,z. Now two options are open: In the first, one can retain $\Delta_2^{(\nu)} x^{\delta}$ as it appears in the equation, and in the second write $\Delta_2^{(\nu)} x^{\delta} = P^{\delta}/G_{\nu}$, where P^{δ} are arbitrarily specified functions. The second option provides a control by the user on the distribution of coordinates in the surface.

II. The proposed equations can be used to generate a new coordinate system from the data of an already given coordinate system in a surface (refs. 10 and 11).

III. If the coefficients of the first and second fundamental forms have been given, then the proposed equations can be used to generate a surface satisfying the given data (surface fitting).

IV. The proposed equations can also be used to generate surfaces in the space between two arbitrary given surfaces, thus providing 3D grids in an Euclidean space (refs. 6, 7, 12, and 13).

A number of numerical and analytical results obtained by the author and his co-workers will be presented in the seminar.

REFERENCES

1. Thompson, J.F.; Warsi, Z.U.A.; Mastin, C.W. Journal of Computational Physics 47, 1982, pp. 1-108.

2. Thompson, J.F. (Ed.), Numerical Grid Generation, North-Holland, 1982.

3. Ghia, K.N. and Ghia, U. (Ed.), Advances in Grid Generation, ASME Publication No. Fed-Vol. 5, 1983.

4. Thompson, J.F.; Warsi, Z.U.A.; and Mastin, C.W. Numerical Grid Generation: Foundations and Applications, North-Holland, 1985.

5. Warsi, Z.U.A., "A Method for the Generation of Three-Dimensional Coordinates Between Bodies of Arbitrary Shapes", MSSU-EIRS-90-7, Mississippi State University, 1980.

6. Warsi, Z.U.A., "Basic Differential Models for Coordinate Generation", in Numerical Grid Generation, Ed. J.F. Thompson, North-Holland, 1982, pp. 41-77.

7. Warsi, Z.U.A., Quart. Appl. Math., 41, 1983, pp. 221-236.

8. Warsi, Z.U.A. and Ziebarth, J.P., "Numerical Generation of Three-Dimensional Coordinates Between Bodies of Arbitrary Shapes", in Numerical Grid Generation, Ed. J.F. Thompson, North-Holland 1982, pp. 717-728.

9. Warsi, Z.U.A., "Numerical Grid Generation in Arbitrary Surfaces Through a Second-Order Differential Geometric Models", Journal of Computational Physics, submitted.

10. Tiarn, W-N., "Generation of Surface Coordinates in Curved Surfaces and in Plane Regions", M.S. Thesis, Mississippi State University, December 1983.

11. Ziebarth, J.P. and Warsi, Z.U.A., "Computer Simulation of Three-Dimensional Grids", Society of Computer Simulation, February 1984.

12. Ziebarth, J.P., "Three-Dimensional Grid Generation Through Elliptic Partial Differential Equations", Ph.D. Dissertation, Mississippi State University, December 1983.

13. Warsi, Z.U.A., "Generation of Three-Dimensional Grids Through Elliptic Differential Equation", CFD Lecture Series, Von Karman Institute for Fluid Dynamics, March 1984.

N85-28584

A SURVEY OF COMPOSITE GRID GENERATION FOR GENERAL THREE-DIMENSIONAL REGIONS

Joe F. Thompson
Department of Aerospace Engineering
Mississippi State University
Mississippi State, MS

The generation and use of composite grids for general three-dimensional physical boundary configurations is discussed, and the availability of several codes or procedures is noted. With the composite framework, the physical region is segmented into sub-regions, each bounded by six curved sides, and a grid is generated in each sub-region. These grids may be joined at the interfaces between the sub-regions with various degrees of continuity. This structure allows codes to be constructed to operate on rectangular blocks in computational space, so that existing solution procedures can be readily incorporated in the construction of codes for general configurations.

Numerical grid generation has become an integral part of the numerical solution of partial differential equations and is one of the pacing items in the development of codes for general configurations. The numerically generated grid frees the computational simulation from restriction to certain boundary shapes and allows general codes to be written in which the boundary shape is specified simply by input. The boundaries may also be in motion, either as specified externally or in response to gradients in the developing physical solution. In any case, the numerically generated grid allows all computation to be done on a fixed square grid in the computational space, which is always rectangular by construction. Boundary conditions can be represented entirely by differences along grid lines without need of interpolation, and hence finite difference methods are readily applicable to general regions. These grids can also serve in finite-element formulations based on quadrilaterals (hexahedrons in 3D), and finite volume constructions can be represented as conservative finite difference forms.

Considerable progress is being made toward the development of the techniques of numerical grid generation and toward casting them in forms that can be readily applied. A comprehensive survey of numerical grid generation procedures and applications thereof through 1981 is given in reference 1, and the conference proceeding published as reference 2 contains a number of expository papers on the area, as well as current results. Other collections of papers on the area have also appeared (refs. 3 and 4), and a later review through 1983 has been given in reference 5. Some other earlier surveys are noted in reference 1. A recent survey by Eiseman is given in reference 6. Surveys particularly on three-dimensional grid generation (refs. 7 and 8) and on adaptive grids (refs. 9 and 10) have also been given. A general text on numerical grid generation and its applications has now appeared (ref. 11).

Since one of the curvilinear coordinates is constant on each segment of the physical boundary, the transformed (computational) field is rectangular with a uniform square grid by construction. The grid is generated from specified grid point distributions and/or grid line intersection angles on the boundaries. The computational region may be an empty rectangular block with all the physical boundary segments corresponding to portions of the sides thereof, or some of these segments may correspond to slits or slabs in the interior of the computational block. Although in principle it is possible to establish a correspondence between any physical region and a single empty rectangular block, for general

PRECEDING PAGE BLANK NOT FILMED

three-dimensional configurations the resulting grid is likely to be much too skewed and irregular to be usable when the boundary geometry is complicated.

A better approach with complicated physical boundaries is to segment the physical region into contiguous sub-regions, each bounded by six curved sides (four in 2D), and each of which transforms to a rectangular block in the computational region, with a grid generated within each sub-region. Each sub-region has its own curvilinear coordinate system, irrespective of that in the adjacent sub-regions. This then allows both the grid generation and numerical solutions on the grid to be constructed to operate in a rectangular computational region, regardless of the shape or complexity of the full physical region. The full region is treated by performing the solution operation in all of the rectangular computational blocks. With the composite framework, partial differential equation numerical solution procedures written to operate on rectangular regions can be incorporated into a code for general configurations in a straightforward manner, since the code only needs to treat a rectangular block. The entire physical field then can be treated in a loop over all the blocks.

The generally curved surfaces bounding the sub-regions in the physical region form internal interfaces across which information must be transferred, i.e., from the sides of one rectangular computational block to those of another. These interfaces occur in pairs, an interface on one block being paired with another on the same or a different block, since both correspond to the same physical surface. The locations of the interfaces between the sub-regions in the physical region are, of course, arbitrary since these interfaces are not actual boundaries. These interfaces might be fixed, i.e., the location completely specified just as in the case of actual boundaries, or might be left to be located by the grid generation procedure. Also, the grid lines in adjacent sub-regions might be made to meet at the interface between with complete continuity, with continuous line slope only, with discontinuous slope, or perhaps not to meet at all. Naturally, progressively more special treatment at the interface will be required in numerical solutions as more degrees of grid line continuity at the interface are lost. Procedures for generating composite grids with these various degrees of continuity at the interfaces are discussed in general in references 1, 2, and 11.

Three-dimensional grid codes should hopefully become suitable for general use in the near future. Further development is now needed in automation of the field segmentation decisions (work on an artificial intelligence approach to this is in progress) and refinement of the geometric procedures for construction of the physical boundaries. The incorporation of dynamically adaptive grids in the composite framework is only just emerging and should prove to be of considerable importance to general flow solutions.

REFERENCES

1. Thompson, Joe F.; Warsi, Z. U. A. and Mastin, C. W., "Boundary-Fitted Coordinate Systems for Numerical Solution of Partial Differential Equations - A Review", Journal of Computational Physics, 47, 1, 1982.

2. Thompson, Joe F. (Ed.), Numerical Grid Generation, North-Holland 1982. (Also published as Vol. 10 and 11 of Applied Mathematics and Computation, 1982).

3. Smith, Robert E., (Ed.), Numerical Grid Generation Techniques, NASA Conference Publication 2166, NASA Langley Research Center, 1980.

4. Ghia, K. N. and Ghia, U.,(Ed.), Advances in Grid Generation, FED-Vol. 5, ASME Applied Mechanics, Bicengineering, and Fluids Engineering Conference, Houston, 1983.

5. Thompson, Joe F., "Grid Generation Techniques in Computational Fluid Dynamics", AIAA Journal, 22, 1505, 1984.

6. Eiseman, Peter R. "Grid Generation for Fluid Mechanics Computations", Annual Review of Fluid Mechanics, 17, 1985.

7. Thompson, J. F. and Warsi, Z. U. A., "Three-Dimensional Grid Generation from Elliptic Systems", AIAA-83-1905, AIAA 6th Computational Fluid Dynamics Conference, Danvers, Mass., 1983.

8. Smith, R. E., "Three-Dimensional Algebraic Grid Generation", AIAA Paper 83-1904, AIAA 6th Computational Fluid Dynamics Conference, Danvers, Mass., 1983.

9. Thompson, Joe F., "A Survey of Dynamically-Adaptive Grids in the Numerical Solution of Partial Differential Equations", Applied Numerical Mathematics, 1, 3, 1985.

10. Anderson, D. A., "Adaptive Mesh Schemes Based on Grid Speeds", AIAA Paper 83-1981, AIAA 6th Computational Fluid Dynamics Conference, Danvers, Mass., 1983.

11. Thompson; Joe F.; Warsi, Z. U. A. and Mastin, C. Wayne, Numerical Grid Generation: Foundations and Applications, North-Holland, 1985.

N85-28585

A UNIFIED REPRESENTATION SCHEME FOR SOLID GEOMETRIC OBJECTS USING B-SPLINES[†]

Extended Abstract

Dennis Bahler
Dept. of Computer Science
University of Virginia
Charlottesville, VA 22903

1. Introduction. A critical task facing those who would construct integrated design systems dealing with solid objects is how best to define and structure the geometric information to achieve maximum flexibility, efficiency, and functionality of the system. In particular, the need arises for an efficient geometric representation scheme capable of representing a broad class of objects in a unified way. In this study we define a geometric representation scheme we call the **B-spline cylinder** [1,2] , which consists of interpolation between pairs of uniform periodic cubic B-spline curves. This approach carries a number of interesting implications. For one, a single relatively simple database schema can be used to represent a reasonably large class of objects, since the spline representation we will describe is flexible enough to allow a large domain of representable objects at very little cost in data complexity. The model is thus very storage-efficient. A second feature of such a system is that it reduces to one the number of routines which the system must support to perform a given operation on objects. Third, the scheme enables easy conversion to and from other representations.

The formal definition of the cylinder entity is given in section 2. In section 3, we explore the geometric properties of the entity, and define several operations on such objects. In section 4 we introduce some general purpose criteria for evaluating *any* geometric representation scheme, and evaluate the B-spline cylinder scheme according to these criteria.

2. The B-Spline Cylinder. A B-spline cylinder is a boundary-determined three-dimensional object defined by $\mathbf{C}(u,v) = (1-v)\,\mathbf{R}_0(u) + v\,\mathbf{R}_1(u)$ where $\mathbf{R}_0(u)$ and $\mathbf{R}_1(u)$ are uniform periodic cubic B-spline curves. It will prove useful for our purposes to think of these curves as consisting of segments. For a curve R, the equation for each segment r_i can be written as

$$\mathbf{r}_i(u) = [u^3\ u^2\ u\ 1]\ \frac{1}{6}\begin{vmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{vmatrix}\begin{vmatrix} \mathbf{p}_{i-1} \\ \mathbf{p}_i \\ \mathbf{p}_{i+1} \\ \mathbf{p}_{i+2} \end{vmatrix} \qquad (1)$$

where the $\mathbf{p}$'s are neighboring vertices of the associated control polygon. Call the segment endpoints, where $u = 0$ above, **nodes**. The linear interpolation to create the cylinder is taken to mean an order-preserving labeling of the nodes as $\alpha_0\beta_n, \alpha_1\beta_{n-1}, \ldots, \alpha_{n-1}\beta_1$ which minimizes $\sum_{i=0}^{n-1}(\alpha_i - \beta_{n-i})^2$. (This criterion corresponds roughly to minimizing the surface area of the resulting object.) An analogous interpolation between corresponding vertices of

[†]This research was supported in part by NASA grant NAG 5-28209, and in part by NSF grant MCS-8302654.

PRECEDING PAGE BLANK NOT FILMED

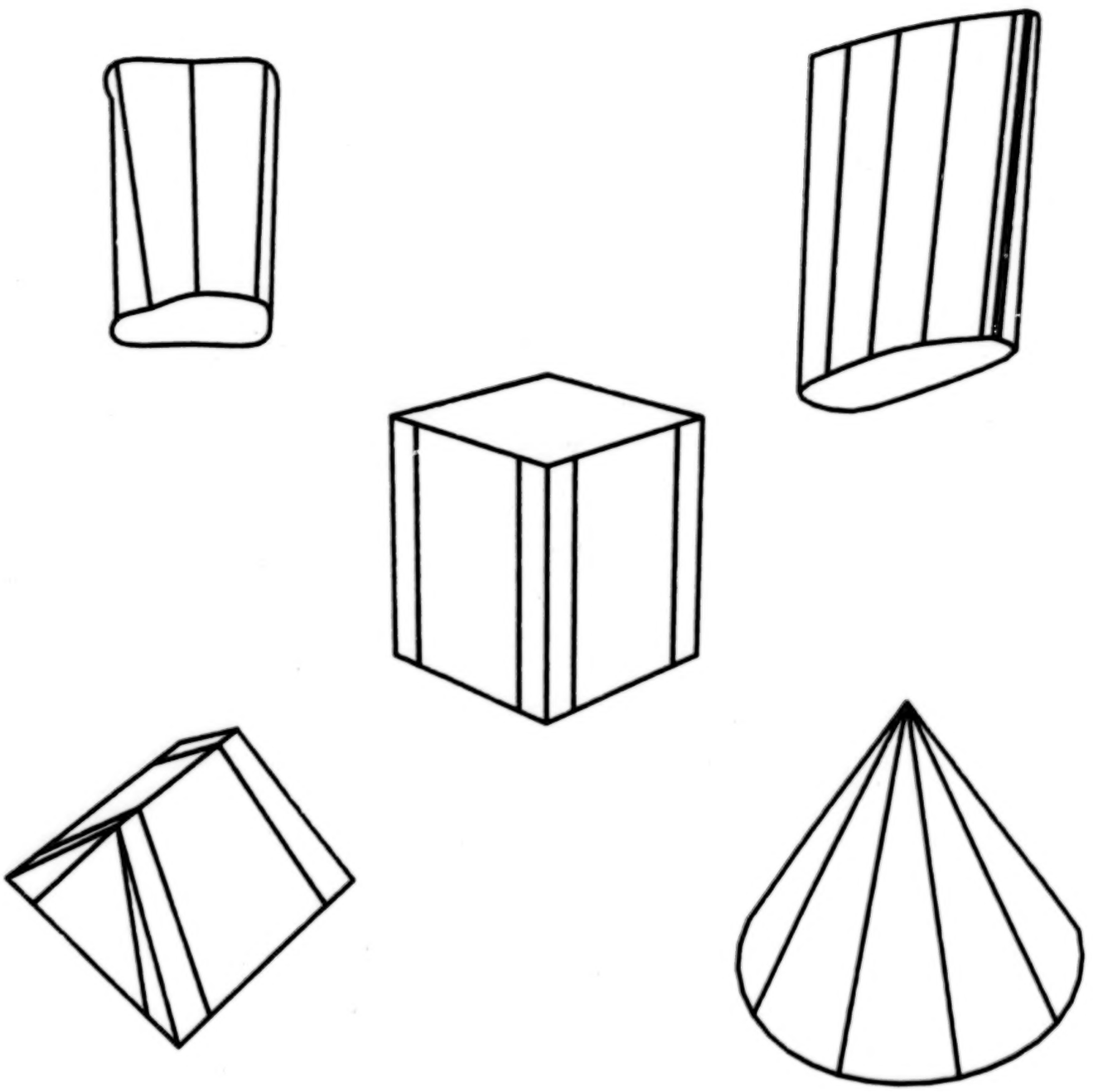


Figure 1. A variety of B-spline cylinders.

the control polygons produces C's **associated prism**. Figure 1 shows the variety of geometric shapes that can be expressed as B-spline cylinders. As can be seen, B-spline cylinders can be used to represent such objects as cubes, regular cylinders, cones, prisms, airfoils, etc. using exactly the same database schema.

3. Operations on Cylinders. It is possible to define a number of unary operators which transform objects into other objects or which yield attributes of objects. Normal vectors, surface area and volumetric properties can be calculated by unary operators. The set membership classification problem for B-spline cylinders (given a point $\mathbf{p}$ and an object C, is $\mathbf{p}$ inside, outside, or on the surface of C?) can be solved using a three-dimensional extension of the angle-sum algorithm for point-in-polygon location.

One way to create new objects from existing ones is to "cut" an object into contiguous smaller objects, which can then be manipulated independently. Refer to Figure 2. The intersection of a B-spline cylinder C with an arbitrary plane P, called a **cut plane**, can be calculated as follows. Define P by the general linear equation $ax + by + cz + d = 0$. Denote the two control polygons defining the faces of C by Q and R. Denote an arbitrary vertex in Q by $\mathbf{q}_i$ and its corresponding vertex in R by $\mathbf{r}_i$. The line segment joining these vertices, which is an edge of C's associated prism, can be defined parametrically by $\mathbf{s}_i = (1 - t_i)\,\mathbf{q}_i + t_i\,\mathbf{r}_i$

These values can be substituted to yield

$$t_i = \frac{aq_{i_x} + bq_{i_y} + cq_{i_z} + d}{a(q_{i_x} - r_{i_x}) + b(q_{i_y} - r_{i_y}) + c(q_{i_z} - r_{i_z})}$$

which can then be used to find the point of intersection $\mathbf{s}_i$ of the cut plane with the line $\mathbf{r}_i - \mathbf{q}_i$. Repeating for all pairs of corresponding vertices in the two faces, a set of intersection points is derived, forming a polygon S in the cut plane. Even in the case of a so-called regular intersection, however, the intersection of the cut plane with C is not always identical to the B-spline curve generated using S as a control polygon. Thus, when a cylinder is segmented with a plane in a regular intersection, it is sometimes not possible to generate the curve of intersection simply by using the polygon resulting from intersecting the associated prism. Fortunately, a polygon can be derived such that it is associated with a curve passing through any set of points, by considering those points as nodes of the curve and solving a simple linear system. The same technique can be used also to derive a control polygon representation for *any* set of points, however arrived at. For example, a curve defined in some other spline formulation, or a set of points obtained from a digitizing device, can be converted into B-spline form. The given points need simply be considered as nodes of a B-spline curve. In this way the B-spline cylinder resembles the generalized cylinder representation used in computer vision. [3]

4. Evaluation of the Cylinder Representation. Defining all objects in the way described carries a number of important advantages, some of which accrue from the properties of B-spline curves themselves, and some from this particular use of them. B-spline curves, and by extension B-spline cylinders, exhibit a number of well-known attractive properties from the standpoint of design. [4] In particular, corners, cusps, and other discontinuities may be introduced easily by using multiple polygon vertices; that is, placing several vertices at the same location. A multiplicity of 3 at two successive vertices will yield an embedded linear segment which is coincident with the corresponding span of the control polygon. It is by this means that objects such as cubes are represented as B-spline cylinders. Moreover, B-spline cylinders are a concise analytic representation for a large class of physical objects, since such objects are determined unambiguously by the control points of the two curves which compose their faces. These control points can be retrieved from a relational database at a cost of $2n + 1$ tuple fetches, where n is the number of vertices in (each of) the two polygons. This can be compared to older spline techniques, for example Hermite

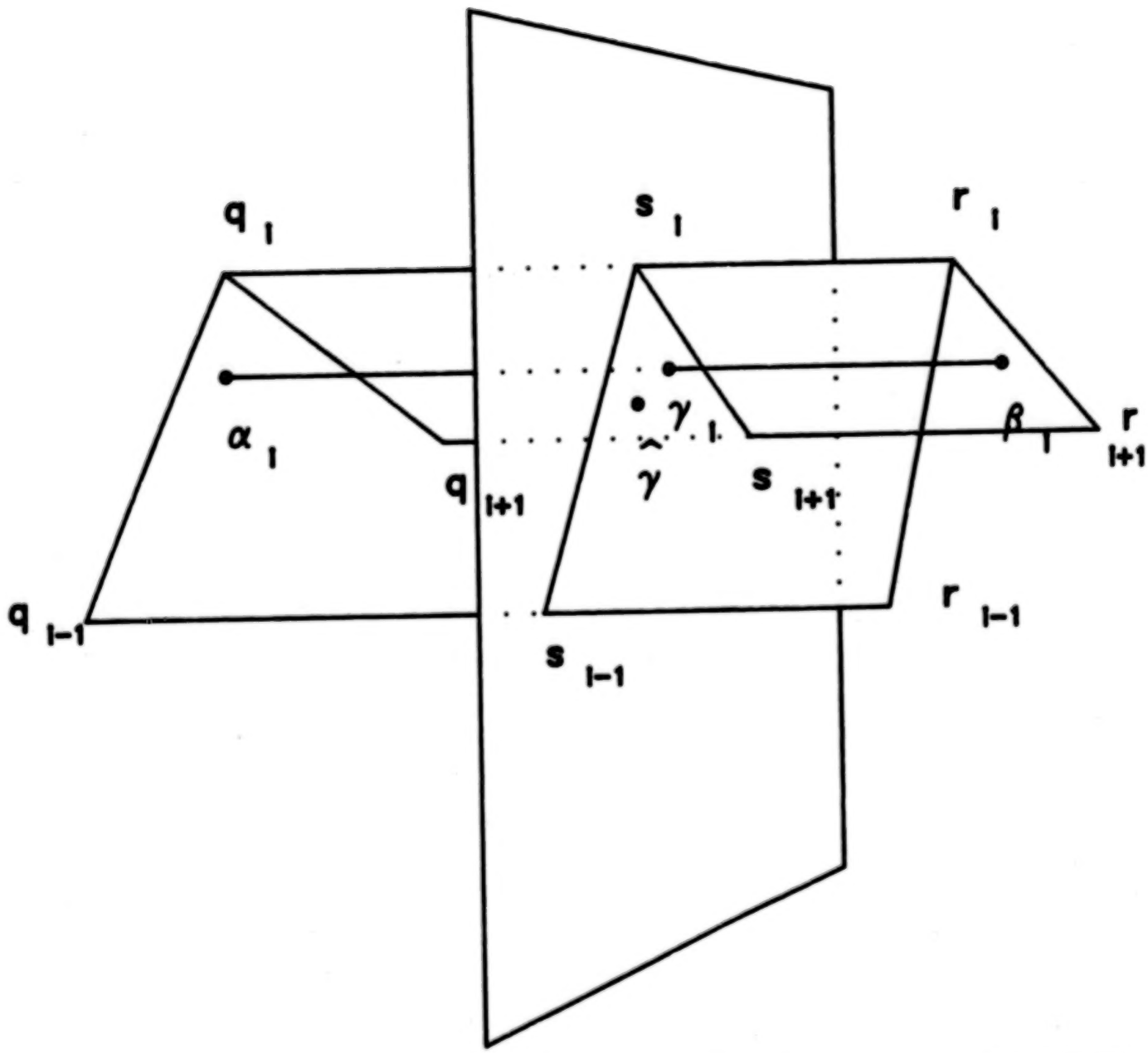


Figure 2. The Plane Cut Problem.

interpolation, which are typically far more verbose.

The B-spline cylinder scheme can also be evaluated with respect to a more formal model of geometric representation, although the definitions and criteria given here are intended to be applicable to any representation scheme. A **representation scheme** S is a mapping $X : M \longrightarrow R$ where M is the set of all suitable subsets of E^3, and R is the set of all syntactically correct terminal strings of some grammar. If $r \in R$ and there exists an $m \in M$ such that $X(m) = r$ then r **represents** or **generates** m through X. The **domain** D of a representation scheme X is the set

$$D = \{m \in M \mid \exists r \in R \text{ and } X(m) = r\}$$

A set $V \subseteq R$ of valid objects is defined by

$$V = \{r \in R \mid \exists d \in D \text{ and } X(d) = r\}$$

where $D \subseteq M$ is the domain of X. Any $r \in V$ is called a **valid** representation. The **range** of a representation scheme is the set V of representations which are valid.

A number of criteria can now be identified by closer examination of the representation scheme mapping X. First, if a representation can correspond to more than one physical object, ambiguity results when the modeling system is called upon to reproduce or reconstruct a previously defined object. A representation $X(m_i)$ of an object $m_i \in D$ is said to be an **unambiguous representation** if for all $m_j \in D$,

$$m_i \neq m_j \longrightarrow X(m_i) \neq X(m_j)$$

that is, X is injective. If for all $v \in V$, v is an unambiguous representation, then X is an **unambiguous representation scheme**. The B-spline cylinder representation scheme is unambiguous; that is, such objects are unambiguously determined by the polygons of their face curves. Second, in a modeling system the intent is to deal as much as possible with representations of objects rather than to have to reconstruct objects explicitly through the mapping X^{-1}. Clearly, it is desirable that an object be representable by one and only one representation; that is, for the representation scheme to be unique. Moreover, uniqueness is necessary in order to draw conclusions about *sets* of objects solely from examination of their representations. One might, for example, wish to detect the congruence of two objects simply by examining their respective representations. If there is a unique $v \in V$ such that $X(d) = v$ then v is a **unique representation** of $d \in D$. X is a **unique representation scheme** if for all $v \in V$, v is a unique representation. Unfortunately, this condition has proven to be extremely difficult to achieve. Virtually no common representation scheme — wire frame, constructive solid geometry, spatial enumeration, oct-trees — is unique. The B-spline cylinder scheme shares this shortcoming.

5. The BCYL System. Many of the ideas discussed in this report have been implemented in a simple geometric modeling and data management system called BCYL. The present version of the BCYL system is written in C and Pascal and runs on a VAX-11/780 under UNIX† 4.2BSD. BCYL employs hidden-surface color graphic display and a relational database management system as a back end. It also contains an editing capability for the interactive creation and modification of objects using a graphics terminal.

6. Conclusions. The definition of what entities and structures are necessary and sufficient for adequately representing geometric information has rarely been addressed. More usually, systems lack coherence and are subject to a combinatorial explosion if system designers attempt to provide users with any capability beyond graphic display. The simplified representation scheme that has been defined in this study and employed in the BCYL system was made for the express purpose of keeping secondary storage accesses to an absolute minimum. All objects in the system are so-called B-spline cylinders. The retrieval of an

† UNIX is a Trademark of Bell Laboratories.

object therefore consists simply of retrieving the control polygon coordinates used to generate its two face curves. This approach appears to be both computationally and storage efficient.

When one is designing a geometry based upon a single representation scheme, there is need for extreme care in selecting that scheme. With this in mind, terminology and concepts have been elaborated by which geometric representation schemes can be described and evaluated by means of a model of the geometric modeling process itself. It has been found that B-spline cylinders possess the essential properties of efficiency and unambiguity and share a common failing among representation schemes in that they lack uniqueness. Moreover, objects defined in this way inherit most of the attractive design characteristics which have made B-spline curves so popular. A B-spline curve can also be interpolated smoothly through an arbitrary collection of data points.

A number of common operations have been defined on B-spline cylinders, the most important of which is regular intersection with a plane. It has been found that the curve of intersection is not in general associated with the polygon of intersection, and an additional operation is necessary on the polygon to re-establish this association. This finding has an important bearing on a system which would permit creation, manipulation, and relinking of cylinders by means of regular intersection.

7. References.

1. Dennis Bahler, "The Representation of Manipulable Solid Objects in a Relational Database," in *Computer-Aided Geometry Modeling*, NASA CP-2272 (1983).

2. Dennis Bahler, "Representation of a Class of Solid Geometric Objects," Tech. Rept. 84-06, Dept. of Computer Science, University of Virginia (August 1984).

3. Rodney A. Brooks, "Symbolic Reasoning Among 3-D Models and 2-D Images," Report No. STAN-CS-81-861, Dept. of Computer Science, Stanford University (June 1981).

4. David F. Rogers and J. Alan Adams, *Mathematical Elements for Computer Graphics*, McGraw-Hill, New York (1976).

N85-28586

IMPROVING THE EFFECTIVENESS OF INTEGRAL PROPERTY CALCULATION IN A CSG SOLID MODELING SYSTEM BY EXPLOITING PREDICTABILITY

Alan L. Clark
Ford Motor Company, Research Staff
Computer-Aided Engineering Department
Dearborn, MI

Integral property calculation is an important application for solid modeling systems. Algorithms for computing integral properties for various solid representation schemes are fairly well known. It is important to designers and users of solid modeling systems to understand the behavior of such algorithms. Specifically the trade-off between execution time and accuracy is critical to effective use of integral property calculation. This paper investigates the average behavior of two algorithms for Constructive Solid Geometry (CSG) representations. Experimental results from the PADL-2 solid modeling system show that coarse decompositions can be used to predict execution time and error estimates for finer decompositions. Exploiting this predictability allows effective use of the algorithms in a solid modeling system.

Integral properties of a solid, sometimes called "mass properties," are defined by a volumetric integral of the form:

$$I = \int_S f(x,y,z)\, dV$$

where the function f is a polynomial, dV is the volume differential, and S is a solid that may be geometrically complex. Thus, for $f(x,y,z) = 1$, the integral represents the volume of the solid. Other functions are used to obtain the centroid, moments of inertia, products of inertia, etc.

There are few algorithms for computing integral properties directly from the representation schemes used in current solid modeling systems. Analytical computation methods for complex solids are extremely difficult. With the exception of polyhedral representations, approximate representation conversion provides the most effective means for computation of integral properties. For solid modeling systems using the CSG representation scheme, the two most effective representation conversion algorithms are column decomposition and block decomposition.

Column decomposition produces a collection of simple elements whose integral properties are trivial to compute. The decomposition is performed using ray tracing techniques. A ray is cast perpendicular to each square from a 2-D grid. Line/solid classification is performed for each ray, producing a columnar decomposition whose elements have a square cross section. Similarly, block decomposition produces a collection of cubical elements whose integral properties are also trivial to compute. The decomposition is performed by recursively subdividing a bounding cube into octants and classifying each octant cube against the CSG solid. The subdivision continues for each octant cube which is neither inside nor outside of the CSG solid, producing a hierarchical decomposition known as an octree. Point/solid classification is performed at the final level of subdivision using a sample point from the octant cube. Using Monte Carlo theory, an error estimate (variance) for the result can be introduced into the algorithms by randomly selecting the target rays and sample points.

The two most important measures of the algorithms' behavior are execution time (a measure of efficiency) and variance estimate (a measure of accuracy). A theoretical analysis of the algorithms only yields results for their worst-case behavior. The worst-case formulas for execution time and variance estimate are based on the level of subdivision used (2-D or 3-D grid size) and the number of primitives in the solid's CSG tree (a rough measure of its complexity). For column decomposition the worst-case execution time is a quadratic function of the number of primitives in the solid's CSG tree and a quadratic function of the number of grid squares along an edge of the solid's bounding cube. For block decomposition the worst-case execution time is a linear function of the number of primitives and a cubic function of the number of minimal size grid cubes along an edge of the bounding cube.

An implementation of the two algorithms can be found in the PADL-2 solid modeling system. Solid models for several automotive mechanical parts were used to generate experimental data for the above two measures of behavior (execution time and variance estimate). An analysis of the data yields a set of formulas for the average behavior of the algorithms that are quite different from the worst-case analysis. For each formula, the variables are the level of subdivision (2-D or 3-D grid size) and a proportionality constant K. This constant K varies from solid to solid and is roughly proportional to the number of primitives in the solid's CSG tree. For column decomposition, the worst-case analysis predicted a quadratic function of the number of primitives, not a linear one. For block decomposition, the worst-case analysis predicted a cubic function of the number of minimal size grid cubes along an edge of the bounding cube, whereas a quadratic function was observed instead.

The most useful fact is that the proportionality constant K is independent of the level of subdivision. A user would like to address questions such as: "How long will it take to compute the integral

properties of a solid to 1% accuracy?" or alternatively "What level of accuracy can be obtained from one hour's execution time?". These questions can be answered by actually computing the integral properties for a subject solid using a low level of subdivision. The values for execution time and variance estimate can be used as "seed" values to compute the K constants for the solid. The formulas will then predict the algorithms' behavior at higher levels of subdivision.

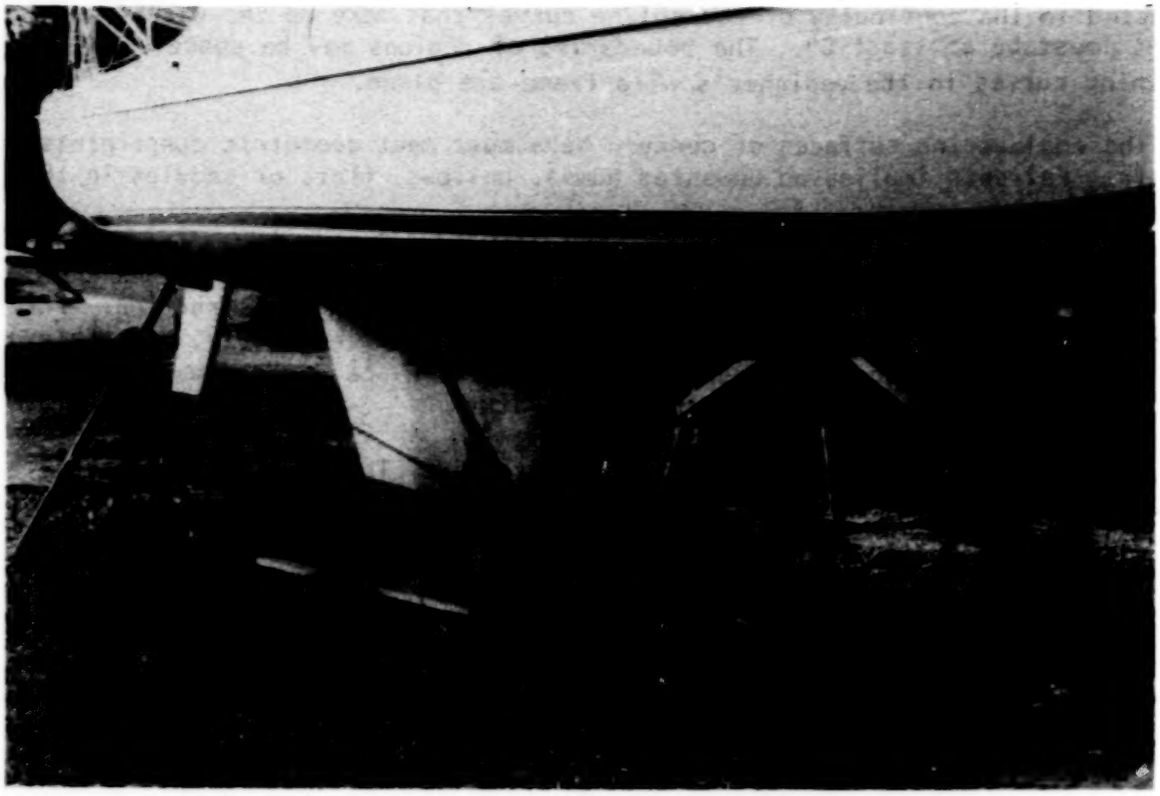
A prediction command has been implemented in the PADL-2 solid modeling system. A comparison of the predicted execution times and variance estimates with actual values shows this prediction cabability accurate enough to be quite useful.

N85-28587

SPLINE CURVES, WIRE FRAMES AND BVALUE

Leigh Smith and Frederick Munchmeyer
University of New Orleans
New Orleans, LA.

ORIGINAL PAGE IS
OF POOR QUALITY



ABSTRACT

Engineers, including naval architects, begin the design of sculptured surfaces by constructing wire-frame curve nets. The boat pictured here first took shape on a drawing board. The wire frame that outlines its shape is called a lines drawing. It is a collection of plane curves drawn with a mechanical spline. Each curve is drawn to fit point, differential, and integral constraints. Curve shape is important.

Modern engineers substitute a terminal for the drawing board and polynomial splines for their mechanical predecessors. Many of them construct their curves with B-splines and find deBoor's FUNCTION BVALUE[1] useful for that purpose. A method centered on BVALUE gives the designer precise control of the fit of each curve to point, differential and integral constraints. It also gives him good control of the shape of each curve and of the whole frame.

PRECEDING PAGE BLANK NOT FILMED

The fundamentals of B-spline construction are well described in the literature. They were first applied to design by Gordon and Riesenfeld[2]. This article presents an extension of the Gordon-Riesenfeld method using BVALUE. The motivation is engineering design, in particular, curved surface design. We use curves constructed with B-splines to create a wire frame and then define a mathematical surface patchwise over that frame. The curves are open in the sense of having free ends; however, one or more elements of the frame may be self intersecting to create a surface that is at least partly closed.

The surface itself may be divisible into regions. Within a region, geodesics and lines of curvature may be required to be continuous which, in turn, implies that the patched surface be at least C^3. The surface continuity requirement is reflected in the continuity of the spline curves that make up the wire frame; the curves must be at least C^4. The boundaries of regions may be space curves; the remaining curves in the designer's wire frame are plane.

The engineering surfaces of concern here must meet geometric constraints and be fair. Fairness implies no unwanted humps, hollows, flats or saddles in the surface.

Some point and differential constraints are defined at regional boundaries. They may be known a priori or they may be determined during construction. For example, the locus of a point on the boundary may be defined early, but the exact distribution of surface curvature along a boundary may be decided during the design process and may be influenced by other constraints.

Data for the surface may not be known directly. They are more likely to be known indirectly as constraints and boundary conditions on a frame that supports the surface. We apply these data to construct the elements of the wire frame, one by one.

The designer first constructs the boundary curves of a region. Each boundary curve meets point, tangent vector, curvature and possibly torsion conditions at its ends. Point constraints may also be imposed along the arc of the curve. If the boundary curve is planar, then an area bounded partially by the curve may be specified. Finally, good distributions of curvature and torsion along boundary curves are important if the surface which they support is to be fair.

The plane curves of the wire frame are designed next, one by one. They define the interior of the region and are labeled s- and t-curves. We emphasize that the initial data require a constructive approach and that they are taken line by line. We have not found an automatic or wholesale method for satisfying the data that guide engineering surfaces. For example, the s-curves may be plane cross sections of an air intake duct which has a prescribed distribution of area along its flow axis. We can form each of the s-curves to satisfy the local area requirement, but we can do so only one at a time.

Some geometric constraints must be met precisely on each s- and t-curve. An elementary but sharp example is the intersection of an s-curve with a t-curve. Measurable error in each of the x-y-z coordinates is allowed at the nominal intersection. The amount allowed is affected by manufacturing tolerance and is likely to be small when compared to the dimensions of the cross section. The designer needs a curve construction method that gives him precise control of the loci of the two curves at their nominal intersection, within a tolerance, and without disturbing the positions of either curve at neighboring mesh points.

The curvature at the end of a wire-frame element contrasts with point constraints along its arc. The curvature of one s-curve at a regional boundary is affected by two wire-frame considerations. First, it must be similar to the end curvatures of its neighboring s-curves. Second, the curvature at one end of a wire-frame element influences shape over much of the arc of that element.

The demand for a fair surface poses a dilemma. We do not know of a design method that guarantees a multi-constraint surface which has no unwanted humps, hollows, flats or saddles. We do not know how to prescribe constraints and boundary conditions that are guaranteed not to conflict with each other in the sense of fairness. Lacking such guidance, we retreat to interactive design. We fair graphs of the boundary derivatives, we fair the graph of cross-section areas, and we fair each s- and t-curve. Fairness in a plane curve implies the absence of unwanted inflections. Further, it means a good shape or a good distribution of curvature; it is largely subjective. The fairness of a graph of boundary derivatives is also subjective.

The main goal of this article is to describe the methods that we have developed for wire-frame design. The principal tools for control of a curve during interactive design are mathematical ducks. The simplest of these devices is an analog of the draftsman's lead weight that he uses to control a mechanical spline. We also create ducks for controlling differential and integral properties of curves.

Other methods presented include:

- constructing the end of a Bézier polygon to gain quick and reasonably confident control of the end tangent vector, end curvature and end torsion

- keeping the magnitude of unwanted curvature oscillations within tolerance

- constructing the railroad curves that appear in many engineering design problems

- controlling the frame to minimize errors at mesh points and to optimize the shapes of the curve elements

REFERENCES

1. Carl deBoor: A Practical Guide to Splines, Springer Verlag, 1978.
2. William J. Gordon and Richard F. Riesenfeld: "B-spline Curves and Surfaces," Computer-Aided Geometric Design, Robert E. Barnhill and Richard F. Riesenfeld, eds., Academic Press, 1974, pp. 95-126.

N85-28588

Numerical Estimation of the Curvature of Biological Surfaces

P.H. Todd
Dept. of Anatomy
Dundee University
DUNDEE
SCOTLAND DD1 4HN

Many biological systems may profitably be studied as surface phenomena. In this paper, we assume a model consisting of isotropic growth of a curved surface from a flat sheet. With such a model, the Gaussian curvature of the final surface determines whether growth rate of the surface is subharmonic or superharmonic. These properties correspond to notions of convexity and concavity, and thus to local excess growth and local deficiency of growth. In biological models where the major factors controlling surface growth are intrinsic to the surface, we have thus gained from geometrical study information on the differential growth undergone by the surface. We look at a specific application of these ideas to an analysis of the folding of the cerebral cortex, a geometrically rather complex surface growth.

The estimation of the curvature of biological surfaces requires a numerical technique which is robust in the presence of noise introduced by whatever digitisation procedure is used to quantify the surface. We develop a numerical surface curvature technique based on an approximation to the Dupin indicatrix of the surface. A metric for comparing curvature estimates is introduced, and considerable numerical testing indicates the reliability of this technique.

The curvatures of normal sections in different directions at a point on a surface are related by a quadratic form called the Dupin indicatrix The eigenvalues of this form correspond to the maximum and minimum curvatures at that point on the surface, the eigenvectors correspond to the directions of maximum and minimum curvature. Gaussian and average curvatures are thus the product and mean respectively of these eigenvalues.

PRECEDING PAGE BLANK NOT FILMED

We estimate surface curvature in the following way. The curvature of a number of sections (not necessary normal) through a point is estimated from triplets of data points taken from a surface digitisation. The normal curvature in the same direction as each section is then estimated using Meusner's theorem. (If K is the normal curvature in the same direction as a section, angle ϕ from normal, and having curvature k, then $K = k \cos \phi$.) A least-squares approximation to the Dupin indicatrix is obtained from the normal curvatures and the principal values and directions of the indicatrix are used as estimates of surface curvature.

For the evaluation of an approximation procedure, it is important to have an appropriate measure of closeness for the quantity or quantities being approximated. A simple norm would suffice to measure Gaussian and average curvatures, which are scalar quantities. A full description of surface curvature, however, involves both the magnitude and direction of the principle curvatures. As these correspond to the eigenvalues and eigenvectors of the Dupin indicatrix, the following definition of a metric for examining surface curvature approximation error seems appropriate. Let L and M be two real symmetric 2 by 2 matrices representing the Dupin indicatrices of two surface curvatures. We define a metric d on the space of such matrices by $d(L,M) = \rho(L-M)$, where $\rho(L-M)$ is the spectral radius of L-M. This is then the spectral norm on the space of real symmetric matrices.

Having defined a suitable metric, we are able to evaluate the accuracy of our curvature estimation procedure in numerical tests using data from known surfaces with varying amounts of imposed noise. We compare the above method with one based on fitting an interpolant surface through points of the digitisation. In the presence of noise, the approach based on the Dupin indicatrix proves the more reliable.

An analysis of the surface curvature of the developing ferret brain reveals that regions of positive and negative Gaussian curvature coincide with the crests of developing folds (gyri) and their troughs (sulci) respectively. In the sense defined by the terms super- and sub-harmonic, gyri correspond to regions of differential excess growth, sulci to regions of differential

growth deficiency. If the underlying growth processes are assumed to be intrinsic to the cortical surface, then we have achieved through analysis of the geometry of the brain a quantitative description of the differential growth of the cortex.

N85-28589

A NEW TECHNIQUE FOR SYSTEM-TO-SYSTEM TRANSFER OF SURFACE DATA

Michael W. Sterling and Michael E. Lucius
CADLINC INCORPORATED
Troy, Michigan

William J. Gordon
Department of Mathematics and Computer Science
Drexel University, Philadelphia, PA

INTRODUCTION

The desirability of transferring data between computer systems has long been recognized. However, due to the diversity of today's CAD/CAM systems, few systems can handle all the currently used mathematical models, nor can new mathematical models be integrated easily into existing software. In particular, a major problem is that of transferring surface geometry information from one system (mathematical model) into another.

Although IGES, for example, allows a number of surface types to be passed, many systems have surface models that do not conform to any IGES standard. Thus, the potential user of IGES is faced with a problem: Can a new surface type be added to IGES, and if so, would all the users of IGES support it?

The purpose of this paper is to describe a recently developed technique aimed at providing a universal interface between surface types. In brief, we have developed a software package which functions as a "common denominator" of CAD/CAM surface types. This software enables one to convert from any given surface representation to any other target representation.

RATIONALE

In most manufacturing environments, surfaces are produced by causing a machine tool to visit an ordered set of points. The surface thus produced is often finished by hand. The finished product may then undergo a quality control check which is designed to insure it matches the theoretical surface to within a specified tolerance. The TILE program mimics this process by producing a set of bicubic patches that matches the original surface in some respects and approximates the surface within a user-specified tolerance.

BASIC CRITERIA

The goal is to produce a set of bicubic patches which cover the target surface in such a fashion that the following criteria are met:

* The number of patches is close to the minimum number necessary

* The patches match the target surface to within some epsilon

* The tiles maintain the same slope continuity as the target surface

The patches produced by the algorithm can be of any type. In the TILE program, we use bicubic patches since they allow us to match point, slope, and twist vectors to the target surface. Thus, slopes can be continuous or discontinuous as they are on the target surface. The patches can be of lower order if desired. For example, if only point information is available, the patches produced will be bilinear; however, the number of patches required is likely to increase correspondingly. The patches can be of higher order although many systems will not accept patches of more than order four.

ALGORITHM

The TILE program assumes the following about the target surface:

* The target surface is defined (or is definable) as a mapping from a rectangle in UV (parameter) space to the surface in XYZ space

* The target surface is divided into a rectangular grid of patches with slope discontinuities occurring only along patch boundaries

* A surface evaluator routine is provided which, given a uv point as input, will compute the surface point, the tangents in the u and v directions, and the twist (mixed partial) vector

If tangent or twist information cannot be obtained, approximations may be computed. However, our experience suggests this will greatly increase the number of patches required to maintain fidelity.

The algorithm begins by attempting to fit a tiling patch to the first patch of the target surface. The target surface evaluator is called for each of the four corners of the patch. The patches generated by TILE all have a unit square as domain. Therefore, the length of the tangent and twist vectors returned by the evaluator must be scaled appropriately. The program assumes the evaluator computes proper lengths. This is not absolutely essential, since there are methods for estimating the correct lengths; however, this may result in an excessive number of patches.

Next the tiling patch is checked for closeness of fit. This is done by comparing the point values of selected points on the edges and interior of the tiling patch with corresponding points on the target surface. If any of these points fail to be within tolerance, then the tiling patch is rejected.

Whenever a tiling patch is rejected, its domain is split into two pieces and new tiling patches are generated and checked for fidelity. How the domain splits (i.e. in the u direction or the v direction) is best determined by examining how the selected points fared in the fidelity test. The domain space is cut along the parameter of worst fit.

The final result of the program is a rectangular grid of bicubic patches. The patches fit the target surface exactly at their corners. Also, the patch corners have the same tangent and twist vectors. Adjacent patches will have slope continuity, unless a discontinuity was indicated by the target surface.

N85-28590

CURVED FINITE ELEMENTS AND CURVE APPROXIMATION

M. Louisa Baart
National Research Institute for Mathematical Sciences
Pretoria

The approximation of parameterized curves by segments of parabolas that pass through the endpoints of each curve segment arises naturally in all quadratic isoparametric transformations. While not as popular as cubics in curve design problems, the use of parabolas allows the introduction of a geometric measure of the discrepancy between given and approximating curves. The free parameters of the parabola may be used to optimize the fit, and constraints that prevent overspill and curve degeneracy are introduced. This leads to a constrained optimization problem in two variables that can be solved quickly and reliably by a simple method that takes advantage of the special structure of the problem.

FORMULATION OF THE PROBLEM

We assume that tessellation has been done in such a way that every curve segment lies to one side of the straight line segment that connects consecutive nodes, and that the approximating curve segment results from a quadratic isoparametric transformation and is therefore parabolic [1] and has the nodes as endpoints. The free parameters of the transformation are the coordinates of that point (α,β) on the approximating parabola where the slope is identical to that of the node-connecting straight-line segment. We impose constraints that prevent image overspill (equivalently, vanishing of the transformation Jacobian over an associated triangular master element) and ensure limit stability as the curve degenerates to a straight line. We introduce coordinate axes in such a way that the nodes are at $(-1,0)$ and $(1,0)$, with the curve segment above the horizontal x-axis. The approximating parabolic segments can then be expressed in parameterized form as

$$x = t + \alpha(1-t^2),\ y = \beta(1-t^2),\ -1 \le t \le 1 \qquad (1)$$

where the parameters α and β completely determine a particular member of the approximating set (1). The axis of the parabola has slope $\frac{\beta}{\alpha}$. The constraints associated with overspill have the form

$$\tfrac{1}{2} - \alpha - \frac{1}{\mu_1}\beta > 0,\ \tfrac{1}{2} + \alpha - \frac{1}{\mu_2}\beta > 0 \qquad (2)$$

where μ_1 and μ_2 are constants that are related to the transformation [2]. We ensure limit stability by restricting the axial slope [1], namely

$$-\frac{1}{m} \le \frac{\alpha}{\beta} \le \frac{1}{m} \qquad (3)$$

where m is a positive stability safety factor. The constraints (2) and (3) define the feasible domain in which the point (α,β) may lie.

The parameters α and β may be used to optimize the approximation of a given curve

$$x = A(s),\ y = B(s),\ -1 \le s \le 1 \tag{4}$$

with

$$A(1) = -A(-1) = 1,\ B(1) = B(-1) = 0,\ B(s) > 0 \text{ for } -1 < s < 1$$

once we have defined a discrepancy measure for (1) and (4). When the approximating curves are parabolas, a uniquely defined distance between any point on the given curve and a parabola can be measured in the direction of the axis of the parabola. This results, in effect, in a re-parameterization of the parabola with respect to the given curve parameter s. Integration of the squares of the distances for the relevant parameter interval $-1 \le s \le 1$ yields the objective function

$$F(\alpha,\beta) = \left(\frac{\alpha^2}{\beta^2} + 1\right) \int_{-1}^{1} (\beta(1 - t^2(s)) - B(s))^2 ds \tag{5}$$

where

$$t(s) = A(s) - \frac{\alpha}{\beta} B(s)$$

This paper is concerned with the optimization of (5) subject to the constraints of (2) and (3).

THE OPTIMIZATION PROCEDURE

When the inverse $\frac{\alpha}{\beta}$ of the axial slope is considered as a new variable γ, (2), (3) and (5) can be expressed in terms of β and γ, and the equivalent constrained optimization problem can be formulated, namely, minimize

$$G(\beta,\gamma) = (\gamma^2 + 1)(\beta^2 C_2(\gamma) + \beta C_1(\gamma) + C_0) \tag{6}$$

where

$$C_2(\gamma) = \int_{-1}^{1} (1 - t^2(s))^2 ds\ ,\ C_1(\gamma) = -2 \int_{-1}^{1} B(s)(1 - t^2(s)) ds$$
$$C_0 = \int_{-1}^{1} B^2(s) ds\ ,\ t(s) = A(s) - \gamma B(s) \tag{7}$$

subject to

$$\beta \ge 0,\ -\frac{1}{m} \le \gamma \le \frac{1}{m},\ \tfrac{1}{2} - \beta\left(\gamma + \frac{1}{\mu_1}\right) > 0,\ \tfrac{1}{2} + \beta\left(\gamma - \frac{1}{\mu_2}\right) > 0 \tag{8}$$

The main advantage of the formulation (6) to (8) is that we can now use a separated optimization procedure [3] to find the best fit. We define the relevant objective function

$$\tilde{G}(\gamma) = \min_{0 \le \beta \le \beta_{max}} G(\beta,\gamma) \tag{9}$$

where β_{max} is a function of γ, and β is restricted to the feasible domain by the upper limit. The function $\tilde{G}(\gamma)$ is optimized with respect to γ, for $-\frac{1}{m} \leq \gamma \leq \frac{1}{m}$, to find the optimal value $\bar{\gamma}$. The optimal $\bar{\beta}$ is obtained by introducing $\bar{\gamma}$ in (9) and optimizing with respect to β. Justification of this procedure is presented in [4], and its implementation is described in [5]. It is reliable, efficient and eminently suitable for use on smaller computers and in the absence of multi-purpose optimization software.

DISCUSSION OF APPLICATIONS

For applications in the field of computer-aided design, the given curves (4) are often cubic polynomials, and the coefficients (7) may be calculated in closed form in terms of the polynomial coefficients by using a symbolic machine language so that families of curves can be approximated with no further integration. For general curves, numerical quadrature may be used, as in the implementation [5] where the Romberg quadrature is applied. The coefficient functions $C_1(\gamma)$ and $C_2(\gamma)$ are expanded as polynomials in γ, so that for given A(s) and B(s) the integrations need only be done once.

The method was used to find optimal constrained parabolic approximation to a wide variety of given curves. Some examples from [6] were included in the numerical tests. A comprehensive discussion of the experimental results is contained in [4]. The method yielded satisfactory approximations to the given curves for all the examples considered.

REFERENCES

1. McLeod, R.J.Y.: Some applications of geometry in numerical analysis. Proceedings of the 9th Biennial Conference on Numerical Analysis, Dundee, 1981. Springer Verlag, Berlin, 1982.

2. Mitchell, A.R.; Wait, R.: The finite element method in partial differential equations. J. Wiley & Sons, London, 1977.

3. Golub, G.H.; Pereyra, V.: The differentiation of pseudo-inverses and nonlinear least squares problems whose variables separate. SIAM J. Numer. Anal. 10, 1973, pp. 413-432.

4. Baart, M.L.; McLeod, R.J.Y.: Constrained parabolic approximation of curves in the finite element method. NRIMS Technical Report TWISK 340, CSIR, Pretoria, December 1983.

5. Baart, M.L.: SPAC: A Fortran subroutine for constrained parabolic approximation of parametric curves. NRIMS Internal Report I521, CSIR, Pretoria, October 1983.

6. Mullineux, G.: Approximating shapes using parameterized curves. IMA. J. Appl. Math. 29, 1982, pp. 203-220.

N85-28591

COMPARISON OF TWO ALGEBRAIC METHODS FOR CURVE/CURVE INTERSECTION

Yves de Montaudouin and Wayne Tiller

Structural Dynamics Research Corporation
Milford , Ohio

Introduction

Most geometric modeling systems use either polynomial or rational functions to represent geometry. In such systems most computational problems can be formulated as systems of polynomials in one or more variables. Classical elimination theory can be used to solve such systems (Refs. 1,2,3,4,5). In this paper we summarize Cayley's method of elimination and show how it can best be used to solve the curve/curve intersection problem.

Summary of Elimination Using Cayley's Method

Let $P(x_1,\ldots,x_n)$, $Q(x_1,\ldots,x_n)$ be polynomials in the n variables $x_1,\ldots,x_n$, and assume that both P and Q are of degree $m>0$ in x_n. If we consider P,Q as polynomials in the one variable x_n , with coefficients which are polynomials in $x_1,\ldots,x_{n-1}$, then we can write

$$P(x_n) = \sum_{k=0}^{m} a_k x_n^k \quad , \quad Q(x_n) = \sum_{k=0}^{m} b_k x_n^k \quad ,$$

where the a_k, b_k are polynomials in $x_1,\ldots,x_{n-1}$.

For $0 \le i,j \le m-1$, $ml=\max(0,i+j-m+1)$, and $m2=\min(i,j)$ define :

$$c_{ij} = \sum_{k=ml}^{m2} a_k b_{i+j-k+1} - a_{i+j-k+1} b_k \, .$$

The matrix $[c_{ij}]$ is called the Cayley (or Bezout) matrix and $\det[c_{ij}]$ is called the resultant of P and Q (denoted by $\mathrm{Res}(P,Q)$).

Theorem : $\mathrm{Res}(P,Q)$ is a polynomial in $x_1,\ldots,x_{n-1}$. The following three conditions are equivalent :

1. $\mathrm{Res}(P,Q) = 0$
2. P and Q have a common root
3. The mxm system of linear homogeneous equations given by $[c_{ij}]\,[x_n^j]^T = 0$ has a solution.

Furthermore , there (usually) exist polynomials $F(x_1,\ldots,x_{n-1})$, $G(x_1,\ldots,x_{n-1})$ such that if $\mathrm{Res}(P,Q) = 0$ for some fixed $x_1,\ldots,x_{n-1}$ then the common root of P and Q is given by $x_n = -F/G$. The functions F and G are derived by Cramer's rule from the system $[c_{ij}]\,[x_n^j]^T = 0$.

PRECEDING PAGE BLANK NOT FILMED

Example.

$P(x) = (x-1)(x-2) = 2-3x+x^2 = 0$
$Q(x) = (x-1)(x+2) = -2+x+x^2 = 0$.

$$[c_{ij}] = \begin{bmatrix} a_0b_1-a_1b_0 & a_0b_2-a_2b_0 \\ a_0b_2-a_2b_0 & a_1b_2-a_2b_1 \end{bmatrix} = \begin{bmatrix} -4 & 4 \\ 4 & -4 \end{bmatrix}$$

Clearly $\det[c_{ij}] = 0$, and from $\begin{bmatrix} -4 & 4 \\ 4 & -4 \end{bmatrix}\begin{bmatrix} 1 \\ x \end{bmatrix} = 0$

it follows that $-4+4x = 0$ and thus $x = 1$.

Curve/Curve Intersection Using Elimination

Let $C_1(s) = (x_1(s),y_1(s))$ and $C_2(t) = (x_2(t),y_2(t))$ be two rational parametric curves , both defined on [0,1]. There are two ways to use elimination to find the intersection points of C_1 and C_2.
Method 1. Implicitize one of the curves.
Steps :
a. Use elimination to convert $C_1(s)$ to its implicit representation $P(x,y) = 0$. In the process we obtain $s = -F(x,y)/G(x,y)$ This is an interesting problem in its own right.
b. Substitute $x_2(t)$, $y_2(t)$ into $P(x,y) = 0$ to get $Q(t) = 0$.
c. Find the roots t_i of $Q(t)$ within the range [0,1].
d. For each t_i , use $x_2(t)$, $y_2(t)$ to obtain the corresponding point (x_i,y_i) on C_2.
e. Use (x_i,y_i) and $s = -F/G$ to get s_i. If s_i is in [0,1] , then s_i , t_i , (x_i,y_i) is a solution (intersection point).

Example.
Let $C_1(s) = ((1-s^2)/(1+s^2),2s/(1+s^2))$ and $C_2(t) = (t,t^2)$. C_1 is a circular , C_2 a parabolic arc , both in the first quadrant.
a. From $x=(1-s^2)/(1+s^2)$ and $y=2s/(1+s^2)$, it follows that
$f(x,y,s) = (x-1)+(x+1)s^2 = 0$
$g(x,y,s) = y-2s+ys^2 = 0$. Eliminating s , we obtain :

$$[c_{ij}] = \begin{bmatrix} -2x+2 & -2y \\ -2y & 2x+2 \end{bmatrix} .$$

Expanding $\det[c_{ij}]$ and setting it equal to zero yields $P(x,y) = x^2+y^2-1 = 0$. And from $(-2x+2)-2ys = 0$, we get $s = (1-x)/y$.
b. Substituting (t,t^2) into $P(x,y)$ yields $Q(t) = t^4+t^2-1 = 0$.
c. The only root in [0,1] is (approx.) $t = 0.786$.
d. From $C_2(t)$, we have $x = 0.786$, $y = 0.618$.
e. From $s = (1-x)/y$ we obtain $s = 0.346$.

Method 2. Subtract the coordinate functions.
Steps :
a. Form $P(s,t) = x_1(s)-x_2(t) = 0$ and $Q(s,t) = y_1(s)-y_2(t) = 0$.

b. Eliminate one of the variables , say s , to get Res(P,Q) = R(t) = 0 and s = -F(t)/G(t).
c. Find the roots t_i of R(t) in the range [0,1].
d. For each t_i , use s = -F/G to get s_i. If s_i is also in [0,1] , use $(x_2(t),y_2(t))$ to get (x_i,y_i).

Example.
Same as above.
a. $0 = P(s,t) = (1-s^2)/(1+s^2) - t = (1-t)-(1+t)s^2$
$0 = Q(s,t) = 2s/(1+s^2) - t^2 = -t^2+2s-t^2s^2$.

b.
$$R(t) = \begin{bmatrix} 2(1-t) & -2t^2 \\ -2t^2 & 2(1+t) \end{bmatrix} = t^4+t^2-1 = 0 .$$

And $2(1-t)-2t^2s = 0$ implies $s = (1-t)/t^2$.
c. Solving for t yields (approx.) t = 0.786 .
d. Substituting into $s = (1-t)/t^2$ yields s = 0.346 .

Comparison of the Two Methods

To our knowledge only method 1 has been mentioned in the CAD/CAM literature (Refs. 2,3). But method 2 is a more straightforward approach. Furthermore , it is computationally simpler , since the elements of the Cayley matrix are one variable instead of two variable polynomials. We implemented and tested both methods and found method 2 to be more efficient. We used six pairs of curves , representing mixtures of lines , circles , and cubic arcs. Several examples had multiple intersection points. For all six cases method 2 required less CPU time than method 1. The average time ratio of method 1 to method 2 was 3.13:1 , the least difference was 2.33:1 , and the most dramatic was 6.25:1 .

Conclusion

Both of the above methods can be extended to solve the surface/surface intersection problem. That is the direction of our current research.

References

1. de Montaudouin , Y. and Tiller , W. , "The Cayley Method in Computer Aided Geometric Design" , Computer Aided Geometric Design, to appear, 1985.
2. Sederberg , T. , Anderson , D. and Goldman , R. , "Implicit Representation of Parametric Curves and Surfaces" , Computer Vision , Graphics and Image Processing , Oct. 1984.
3. Sederberg , T. , Anderson , D. and Goldman , R. , "Implicitization , Inversion , and Intersection of Rational Cubic Curves" , Computer Aided Geometric Design, to appear, 1985.
4. Kajiya , J. , "Ray Tracing Parametric Patches" , Computer Graphics , Vol. 16 , No. 3 , 1982.
5. Salmon , G. , "Modern Higher Algebra" , Hodges, Smith and Co., Dublin , 1866.

N85-28592

A Comparison of Three Curve Intersection Algorithms

Thomas W. Sederberg
Scott R. Parry
Brigham Young University
Provo, Utah 84602

EXTENDED ABSTRACT

An empirical comparison is made between three algorithms for computing the points of intersection of two planar Bezier curves. The algorithms compared are:

A1. The well known Bezier subdivision algorithm, which is discussed in [Lane'80]

A2. A subdivision algorithm based on interval analysis due to Koparkar and Mudur [Koparkar'83]

A3. An algorithm due to Sederberg, Anderson and Goldman which reduces the problem to one of finding the roots of a univariate polynomial [Sederberg'84].

The details of these three algorithms are presented in their respective references. This abstract mentions major implementation choices, and assumes that the reader is familiar with the three algorithms.

The execution times of A1 and A2 are highly sensitive to implementation. We observed that the decision on when to terminate the subdivision can effect execution speed by an order of magnitude. We based our termination criterion on a suggestion from [Wang'84] that a curve be subdivided until its deviation from a straight line segment is no larger than a value ϵ. If (x_i,y_i) $i = 0,1,\ldots,n$ are the control points of a degree n Bezier curve and

$$L_0 = \max_{0\le i\le n-2}(|x_i - 2x_{i+1} + x_{i+2}|,|y_i - 2y_{i+1} + y_{i+2}|),$$

then

$$r_0 = \log_4 \frac{\sqrt{3}n(n-1)L_0}{8\epsilon}$$

where r_0 is the number of times the curve must be subdivided in order for a curve segment to approximate a line segment to within a tolerance ϵ.

PRECEDING PAGE BLANK NOT FILMED

For A1 and A2, the convex hull was taken to be a bounding rectangle. For A1, subdivision was performed using the geometric construction algorithm and the bounding rectangle was determined by the new Bezier control points. For A2, the polynomial equations were converted to standard power basis, and subdivision was performed by simply evaluating the curve segment at its parametric midpoint, using Horner's algorithm.

For A3, the implicitization procedure was performed using Bernstein polynomials. It was noted that standard polynomials introduced excessive roundoff error for curves of degree larger than three. A critical part of the implementation is a polynomial root finder which finds only real roots. We developed our own root finder for this purpose, which usually runs significantly faster than, for example, the standard IMSL root finder which finds *all* roots (real and complex) of a polynomial.

We tested these three algorithms on pairs of Bezier curves of degree three, four, and five, with several different numbers of intersection points. Our goal was to compute the intersection points to five decimal places of accuracy, and we experimented to find the optimal value of r_0 for this purpose.

Results: The tests were run on a VAX11/750, with floating point accelerator, under UNIX 4.2 BSD. Our test results indicate that for two cubic Bezier curves which intersect once, A3 is at least twice as fast as the other two algorithms. For more than one intersection, A3 does even better -- if the two cubic curves intersect nine times, A3 is typically five or six times faster than the others.

For curves of degree four, A2 and A3 are comparable, and A1 is the slowest. For curves of degree five, A2 is typically twice as fast as the other two algorithms.

Multiple intersections were not thoroughly tested. Our implementation of A1 and A2 did not properly detect the presence of tangency. A3 handled tangency with no problem by reporting a multiple polynomial root.

References

Koparkar, P.A. and Mudur, S.P.(1983), "A new class of algorithms for processing parametric curves", *Computer-Aided Design*, Vol. 15, No. 1, 41-45.

Lane, J.M., and Riesenfeld, R.F.(1980), "A theoretical Development for the Computer Generation and Display of Piecewise Polynomial Surfaces", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. PAMI, No. 1,35-46.

Sederberg, T.W., Anderson, D.C. and Goldman, R.N.(1984), "Implicit representation of parametric curves and surfaces", *Computer Vision, Graphics and Image Processing*, Vol. 28, 72-84.

Wang, G. (1984), "The Subdivision Method for Finding the Intersection between Two Bezier Curves or Surfaces", *Zhejiang University Journal*, Special Issue on Computational Geometry.

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
NASA CP-2379		

4. Title and Subtitle	5. Report Date
Computational Geometry and Computer-Aided Design	June 1985
	6. Performing Organization Code: 505-31-83-02

7. Author(s)	8. Performing Organization Report No.
Temple H. Fay and John N. Shoosmith, compilers	L-16003
	10. Work Unit No.

9. Performing Organization Name and Address	11. Contract or Grant No.
NASA Langley Research Center Hampton, VA 23665	
	13. Type of Report and Period Covered: Conference Publication

12. Sponsoring Agency Name and Address	14. Sponsoring Agency Code
National Aeronautics and Space Administration Washington, DC 20546 and University of Southern Mississippi Hattiesburg, MS 39406	

15. Supplementary Notes

16. Abstract

The Conference on Computational Geometry and Computer-Aided Design was held in New Orleans, Louisiana, on June 5-8, 1985. This conference provided both invited and submitted papers in such areas as the numerical representation of curves and surfaces, solid geometry modeling, management of geometric data, and development of geometric standards, with the focus on geometric and mathematical aspects rather than on software aspects. The goal of the conference was to provide a forum where researchers and implementers from government, academia, and industry could discuss state-of-the-art developments and suggest problems and directions for future research. The conference was organized into sessions on the following topics: (1) curves and shape control, (2) surfaces, (3) grid generation and contouring, (4) solid modeling, and (5) curve intersections. These proceedings present extended abstracts for the papers presented at the conference, including the nine invited addresses.

17. Key Words (Suggested by Author(s))	18. Distribution Statement
Computer-aided design Computer-aided modeling Numerical modeling Interactive computer modeling Geometry modeling Geometric data management	Unclassified - Unlimited Subject Category 59

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price
Unclassified	Unclassified	135	A07

For sale by the National Technical Information Service, Springfield, Virginia 22161 NASA-Langley, 1985